UNIVERSITÉ DU QUÉBEC À MONTRÉAL

L'APPARIEMENT DE SCHÉMAS COMME UN SYSTÈME COMPLEXE ADAPTATIF : UNE NOUVELLE APPROCHE BASÉE SUR LA MODÉLISATION ET LA SIMULATION À BASE D'AGENTS

THÈSE

PRÉSENTÉE

COMME EXIGENCE PARTIELLE

DU DOCTORAT EN INFORMATIQUE

PAR

HICHAM ASSOUDI

AVRIL 2017

REMERCIEMENTS

Tout d'abord, je tiens à remercier et à exprimer toute ma gratitude auprès de mon directeur de thèse Hakim Lounis, Professeur et Directeur du Doctorat en Informatique Cognitive à l'Université du Québec à Montréal, qui grâce à son encadrement intelligent, rigoureux et flexible à la fois, m'a permis, malgré les difficultés, de progresser, tout au long de ces années de recherche, avec un haut degré de confiance et d'autonomie. Je lui suis également reconnaissant de m'avoir toujours accordé, avec une grande générosité, sa disponibilité, son expertise et ses conseils inestimables. Je remercie également les membres du jury pour l'honneur qu'ils m'ont accordé en acceptant d'évaluer cette thèse.

Sur un plan plus personnel, mes premiers remerciements s'adressent à mes parents adorés, qui ont su m'inculquer dès mon jeune âge le goût du dépassement, pour leur amour inconditionnel, leur confiance et leurs perpétuels encouragements. J'adresse également des remerciements à ma chère épouse pour son soutien indéfectible et qui a su trouver les mots qu'il fallait dire, pendant les périodes de doute, pour me permettre de me recentrer, remettre le pied à l'étrier et continuer d'avancer. Je ne saurai passer sous silence le soutien chaleureux de mes frères et sœurs, dont la fierté, qu'ils n'ont eu cesse de montrer à mon égard, a été une grande source d'inspiration. Un clin d'œil de connivence à mon petit frère, informaticien, pour n'avoir jamais montré la moindre lassitude de nos discussions concernant mes travaux de recherche.

Je ne saurais terminer sans remercier toutes les personnes qui ont, chacune à leur façon, et ce, à différentes étapes de mon cheminement, contribué, d'une manière ou d'une autre, à la réalisation de cette thèse de doctorat.

TABLE DES MATIÈRES

LISTE DES FIGURES IX
LISTE DES TABLEAUX XIII
RÉSUMÉ XV
INTRODUCTION 1
0.1 Problématique et motivation 1
0.2 Approche et contributions 5
0.3 Organisation de la thèse 9
0.4 Publications 10
CHAPITRE I 13
ÉTAT DE L'ART 13
1.1 Introduction 13
1.2 Appariement de schémas 14
1.2.1 Préliminaires 14
1.2.2 Processus d'appariement 23
1.2.3 Mesures de similarité 31
1.2.4 Défis et difficultés 35
1.2.5 Synthèse 45
1.3 Réductionnisme des approches existantes 48
1.3.1 Complication 50
1.3.2 Linéarité 52
1.3.3 Centralisation 54
1.3.4 Non-adaptation 56
1.3.5 Discussion 57
1.4 Systèmes complexes adaptatifs 59
1.4.1 Concepts 59

1.4.2 Caractéristiques....63
1.4.3 Modélisation et simulation à base d'agents....66
1.4.4 Modélisation de la prise de décision sous l'incertitude....73
1.5 Conclusion....77
CHAPITRE II....79
APPARIEMENT DES SCHÉMAS COMME UN SYSTÈME COMPLEXE ADAPTATIF....79
2.1 Introduction....79
2.2 Cadre théorique....80
2.3 Cheminement méthodologique....89
2.4 Modélisation et simulation à base d'agents pour l'appariement de schémas (SMAS)....93
2.4.1 Exemple de référence....93
2.4.2 Formulation du modèle conceptuel....95
2.4.3 Transcription du modèle....113
2.5 Conclusion....114
CHAPITRE III....115
SIMULATION D'AGENTS RÉFLEXIFS (REFLEX-SMAS)....115
3.1 Introduction....115
3.2 Modèle opérationnel....116
3.2.1 Agent élément de schéma (*SE-Agent*)....119
3.2.2 Relations....125
3.2.3 Plateformes....127
3.3 Prototype et expérimentations....129
3.3.1 Prototype....130
3.3.2 Expérimentations....141
3.4 Collecte et exploitation des données....162
3.4.1 Réseau des interactions....162
3.4.2 Apparieur prédictif....163
3.5 Conclusion....168
CHAPITRE IV....171
VÉRIFICATION ET VALIDATION DU PROTOTYPE REFLEX-SMAS....171

4.1 Introduction 171
4.2 Objectifs et stratégie de la validation 171
4.3 Expérimentations 175
4.3.1 Méthodologie 175
4.3.2 Configuration 181
4.3.3 Résultats 183
4.4 Synthèse et discussion des résultats 192
4.5 Conclusion 205
CHAPITRE V 207
VERS UNE SIMULATION D'AGENTS RATIONNELS (RATIONAL-SMAS) 207
5.1 Introduction 207
5.2 Agent Rationnel (*SEAgentRational*) 207
5.2.1 Modèle conceptuel 208
5.2.2 Architecture et implémentation 216
5.3 Conclusion 218
CONCLUSION 219
BIBLIOGRAPHIE I

LISTE DES FIGURES

Figure Page

Figure 1.1 Exemple de schémas pour l'appariement (Rahm et Bernstein, 2001a) .. 14
Figure 1.2 Processus de l'appariement (Shvaiko et Euzenat, 2005) 16
Figure 1.3 Classification des techniques classiques (Rahm et Bernstein, 2001a).... 20
Figure 1.4 Processus général d'appariement de schémas (Rahm, 2011) 24
Figure 1.5 Topologies des processus d'appariement (Peukert et al., 2012)..... 25
Figure 1.6 Processus d'appariement (Peukert et al., 2010) 27
Figure 1.7 Causes et effets relatifs à l'appariement 47
Figure 1.8 Processus d'appariement (Bellahsene et Duchateau, 2011)..... 49
Figure 1.9 « Complexe » vs. « Compliqué » (Jones, 2003) 61
Figure 1.10 Carte de sciences de la complexité (Castellani, 2014a, 2014b) 63
Figure 1.11 Systèmes Complexes Adaptatifs (Andrus, 2005) 64
Figure 1.12 Taxonomie de la simulation (Fishwick, s.d.) 68
Figure 1.13 Structure d'un agent (Macal et North, 2010) 70
Figure 1.14 Exemple de diagramme d'influence (Matsumoto et al., 2011)..... 76
Figure 2.1 Le système d'appariement de schémas et ces composants 82
Figure 2.2 États du système complexe de l'appariement de schémas..... 86
Figure 2.3 La Matrice de Stacey (Zimmerman, 2001) 88
Figure 2.4 Cheminement méthodologique 92
Figure 2.5 Exemple de référence pour l'appariement 93
Figure 2.6 Processus d'appariement pour le scénario exemple de référence 94
Figure 2.7 Simulation multi-agents pour l'appariement de schémas 97
Figure 2.8 Représentation d'un « agent élément de schéma » (SE-Agent)..... 101
Figure 2.9 Appariement stochastique..... 104

Figure 2.10 Appariement consensuel 107
Figure 2.11 Résolution du problème d'appariement par émergence 109
Figure 2.12 Analyse statistique sur le résultat de plusieurs simulations 112
Figure 2.13 Évolution de l'outil SMAS 113
Figure 3.1 C'est quoi un « Agent » (Macal et North, 2006) 117
Figure 3.2 Plateformes de simulation multi-agents (Macal et North, 2006) 128
Figure 3.3 Architecture du prototype SMAS 131
Figure 3.4 Diagramme de classes pour Reflex-SMAS 133
Figure 3.5 Diagramme d'activités du comportement de l'agent SEAgentReflex ..136
Figure 3.6 Transition entre états de l'agent 137
Figure 3.7 Prototype SMAS exécuté sur la plateforme Repast Simphony 139
Figure 3.8 Panneau de visualisation au début de la simulation 140
Figure 3.9 Panneau de visualisation à la fin de la simulation 141
Figure 3.10 Test du scénario « exemple de référence » avec l'outil COMA 145
Figure 3.11 Interface JoSQL pour l'interrogation de l'état de la simulation 147
Figure 3.12 Aperçu de la simulation a l'itération 11 148
Figure 3.13 Différents aperçus de la simulation 150
Figure 3.14 Visualisation du résultat final (itération 1048) 151
Figure 3.15 Nombre de calculs de similarité lors d'une simulation 154
Figure 3.16 Interface d'exécution des simulations en lot 157
Figure 3.17 Résultat de multiples simulations 158
Figure 3.18 Tableau de contingence des multiples simulations 159
Figure 3.19 Carte thermique du résultat d'une simulation en lot 160
Figure 3.20 Analyse statistique avec l'outil R 161
Figure 3.21 Réseau des interactions 163
Figure 3.22 Exemples étiquetés avec l'outil Weka 167
Figure 4.1 Scénario d'appariement « Person » 176
Figure 4.2 Scénario d'appariement « Order » 176
Figure 4.3 Scénario d'appariement « Travel » 177

Figure 4.4 Résultats de l'évaluation des scénarios avec l'outil COMA 178
Figure 4.5 Résultat de la méta-simulation n°1 pour le scénario « Person » 183
Figure 4.6 Précision, Rappel et F-mesure pour le scénario « Person » 186
Figure 4.7 Résultat de la méta-simulation n°2 pour le scénario « Order » 187
Figure 4.8 Précision, Rappel et F-mesure pour le scénario « Order » 189
Figure 4.9 Résultat de la méta-simulation n°3 pour le scénario « Travel » 190
Figure 4.10 Précision, Rappel et F-mesure pour le scénario « Travel » 192
Figure 4.11 Précision, Rappel et F-Mesure pour tous les scénarios 194
Figure 4.12 Comparaison des résultats pour tous les scénarios 195
Figure 4.13 Impact du nombre de simulations sur la performance 197
Figure 4.14 Impact du nombre d'itérations d'une simulation sur la performance . 198
Figure 4.15 Temps de réponse pour tous les scénarios 200
Figure 4.16 Impact du nombre des simulations le temps total d'exécution 201
Figure 4.17 Impact du nombre d'instances sur le temps d'exécution 202
Figure 4.18 Performances de Reflex-SMAS par rapport à l'outil COMA 204
Figure 5.1 Agent à base d'utilité (Russell et al., 2010) 209
Figure 5.2 Diagramme d'influence pour l'agent SEAgentRational 211
Figure 5.3 Propagation des croyances pour l'appariement candidat 215
Figure 5.4 Propagation des croyances pour l'appariement consensuel 216
Figure 5.5 Diagramme d'activités pour l'agent SEAgentRational 217

LISTE DES TABLEAUX

Tableau		Page
Tableau 1.1	Principales étapes du processus de l'appariement de schémas	31
Tableau 1.2	Objectifs finaux de la simulation (Treuil et al., 2008)	69
Tableau 2.1	Caractéristiques de « Agent Élément de Schéma » (SE-Agent)	100
Tableau 3.1	Sélection des attributs de la classification	165
Tableau 4.1	Tableau des contingences pour les mesures de performance	180
Tableau 4.2	Spécification matérielle	182
Tableau 4.3	Mesure des performances pour le scénario « Person »	184
Tableau 4.4	Mesure des performances pour le scénario « Order »	188
Tableau 4.5	Mesure des performances pour le scénario « Travel »	191
Tableau 4.6	Résultats combinés pour tous les scénarios	193
Tableau 4.7	Résultat combiné pour la qualité de l'alignement trouvé	195
Tableau 4.8	Performances de Refle-SMAS par rapport à l'outil COMA	203

RÉSUMÉ

L'appariement automatique de schémas est une tâche complexe à plus d'un titre : (i) l'hétérogénéité et l'ambiguïté intrinsèque aux éléments de schémas à apparier, (ii) le caractère incertain des résultats de l'appariement, (iii) le défi que peut poser l'optimisation de l'appariement (explosion combinatoire), etc.

Dans le cadre de notre recherche nous avons investigué le recours à la théorie des *systèmes complexes adaptatifs*, issues de *la pensée systémique*, pour chercher, loin des sentiers battus, des réponses innovantes aux défis auxquels les approches classiques d'appariement automatique de schémas, font toujours face (e.g. complexité, incertitude).

Ainsi, nous proposons un modèle conceptuel de *simulation multi-agents pour l'appariement automatique de schémas* (*SMAS*), découlant du domaine des *systèmes complexe adaptatifs*. La transcription de ce modèle conceptuel, cristallisant les principes phares de la théorie des systèmes complexes adaptatifs, notamment la *non-linéarité*, *la stochasticité*, *l'auto-organisation* et *l'émergence,* a donné lieu à un prototype pour l'appariement automatique de schémas, que nous avons baptisé *Reflex-SMAS*. Ce dernier a été soumis à une série d'expérimentations dont l'objectif était de démontrer la viabilité de notre approche relativement à deux aspects principaux : (i) *l'efficacité* (augmentation de la qualité de l'alignement trouvé) et (ii) *l'efficience* (réduction de l'effort nécessaire à cette efficacité). Les résultats obtenus, sont venus apporter la preuve de la viabilité de notre approche, que ce soit sur le plan de l'efficacité ou encore sur celui de l'efficience.

Avec la preuve apportée de la viabilité, nous pouvons désormais annoncer la naissance d'un nouvel outil, dans le domaine de l'appariement automatique de schémas, représentant un changement de paradigme significatif dans ce domaine (au meilleur de nos connaissances, jamais l'appariement automatique de schémas n'a été abordé en adoptant la pensée systémique (holistique), en le considérant comme un système complexe adaptatif et en le modélisant comme une simulation multi-agents).

Mots-clés

Appariement des schémas, systèmes complexes adaptatifs, modélisation et simulation à base d'agents, apprentissage automatique, réseaux bayésiens

INTRODUCTION

0.1 Problématique et motivation

Une des tâches clés dans le développement des solutions permettant l'interopérabilité entre des systèmes d'information hétérogènes, est l'appariement des schémas « *Schema Matching* ». En effet, cette dernière est omniprésente dans plusieurs champs d'application, impliquant la gestion des métadonnées (i.e. schémas, ontologies). C'est le cas de l'intégration, l'échange ou la migration des données, le web sémantique, le commerce électronique, etc. (Rahm et Bernstein, 2001a ; Shvaiko et Euzenat, 2005). La tâche de l'appariement des schémas vise à trouver des relations de correspondances sémantiques entre les éléments des schémas de données (Rahm, 2011).

Pendant longtemps, cette tâche est demeurée une tâche manuelle réservée principalement aux experts ayant une bonne compréhension de la sémantique des différents schémas, plus une maitrise des langages de transformation. En revanche, à mesure que les schémas devenaient plus complexes, cette tâche a commencé à devenir une tâche fastidieuse, chronophage et source d'erreurs (Bellahsene Bonifati Duchateau *et al.*, 2011 ; Bonifati *et al.*, 2011). Dès lors, depuis de nombreuses années, plusieurs recherches se sont penchées sur la problématique de l'automatisation de la tâche de l'appariement de schémas (incluant l'appariement des ontologies[1]). Le but principal visé étant le développement de techniques (algorithmes, outils, etc.) permettant la découverte automatique ou semi-automatique des correspondances entre les éléments

[1] Les disciplines de l'appariement des schémas et des ontologies, partagent dans une large mesure, les mêmes problématiques, approches et techniques (Bellahsene Bonifati Duchateau *et al.*, 2011). Bien que cette thèse traite principalement de l'appariement de schémas de données, il n'en demeure pas moins que l'approche proposée puisse être appliquée au traitement de l'appariement des ontologies.

des différents schémas. Durant la dernière décennie, plusieurs outils s'appuyant sur différentes approches, et permettant l'automatisation de l'appariement de schémas, ont vu le jour (Shvaiko et Euzenat, 2005 ; Bernstein *et al.*, 2011). Certains d'entre eux ont même évolué du cadre de la recherche académique vers l'industrie. *Clio*, dont le prototype a démarré comme projet de recherche conjoint entre *IBM* et *l'université de Toronto*, en est un bon exemple (Miller *et al.*, 2001).

Avec la multiplication des recherches et des outils, une étape importante vers la réalisation de la vision de l'appariement automatisé des schémas a été franchie. Il n'en demeure pas moins que l'incertitude, inhérente à la tâche de l'appariement des schémas, continue à rendre indispensable l'implication de l'humain de sorte qu'elle constitue aujourd'hui un véritable frein à cette vision d'automatisation complète. La raison de cette incertitude réside principalement dans l'ambiguïté et l'hétérogénéité, tant au niveau syntaxique que sémantique, qui peut caractériser la description des éléments des schémas à apparier. Par voie de conséquence, la gestion de l'incertitude dans le processus d'appariement des schémas reste une question de recherche ouverte qui a motivé plusieurs travaux de recherche (Madhavan *et al.*, 2002 ; Gal *et al.*, 2005 ; Gal, 2006b, 2006a ; Nottelmann et Straccia, 2007 ; Castano *et al.*, 2008 ; Gal, 2011b ; Gong *et al.*, 2012 ; Nian-Feng et Xing-Chun, 2012 ; Rizopoulos, 2010 ; Zhang *et al.*, 2013).

Outre le défi de l'incertitude, la multiplication des approches proposées pour le processus de l'appariement témoigne de la complexité de ce processus. Cette complexité est à l'origine d'autres difficultés, comme l'optimisation des performances du processus d'appariement. En plus de la validation du résultat final de ce processus, l'implication humaine est souvent requise aussi pour l'optimisation (e.g. sélection des mesures de similarité, sélection et combinaison des appariеurs) ou le calibrage des paramètres spécifiques à certains appariеurs (e.g. formules, seuils, poids). Cette tâche difficile demande beaucoup d'expertise car les valeurs et les différentes combinaisons

de ces paramètres ont un effet direct sur la performance du processus. Malheureusement, l'effet sur la performance (e.g. qualité des correspondances trouvées) ne peut être mesuré que par un exercice d'essais-erreurs (i.e. expérimentations). Cet exercice peut non seulement être fastidieux mais aussi, peut se révéler peu efficace car souvent l'expert ne peut pas explorer toutes les possibilités (espace de recherche trop large). C'est pour cette raison que plusieurs approches permettant l'automatisation de cette tâche d'optimisation ont été proposées dans la littérature (Lee *et al.*, 2007 ; Bellahsene et Duchateau, 2011).

Nous remarquons toutefois, que les approches proposées, que ce soit pour le processus d'appariement automatique lui-même ou pour le processus qui permet l'optimisation automatique de ces performances, repose largement sur l'interaction humaine immédiate pour la validation du résultat du processus, dans la phase post-appariement, ou pour la configuration/optimisation du processus durant la phase pré-appariement. Certes, cet interaction humaine est généralement envisageable pour de nombreux contextes; par contre dans d'autres contextes où les environnements sont hautement dynamiques (e.g. web sémantique, composition de services web, communication entre les agents), l'implication d'expert doit être réduite au minimum (Cross, 2003).

En somme, force est de constater que la multiplication des approches pour la problématique de l'appariement automatique de schémas, dénote non seulement de la profondeur des questions encore ouvertes de cette problématique, mais aussi de la difficulté de proposer une approche générique capable de répondre, ou à tout le moins, d'aborder toutes ces questions. Effectivement, la majorité de ces approches sont réductionnistes dans le sens où elles essayent d'apporter des réponses ciblant des questions spécifiques avec une approche qui elle tend à découper le tout en parties indépendantes et ensuite à optimiser une de ces partie (e.g. sélection des appariеurs, combinaison des résultats, sélection des paires).

Ainsi, nous avons été motivés à investiguer d'autres paradigmes pour approcher la problématique de l'appariement automatique de schémas, comme un sujet complexe approprié pour une approche holistique (systémique) : une approche, qui permettrait de voir cette problématique comme un tout, c'est-à-dire, comme un système dont l'exécution, la configuration et l'optimisation sont tributaires non seulement des différents composants du système, mais aussi sont basées sur la prise en compte des relations et interactions entre ces composants.

Compte tenu de ces considérations, notre recherche s'est tournée vers l'exploration de nouvelles avenues, dans l'optique de tenter d'apporter une réponse à la question générale suivante :

- *Serait-il possible d'aborder la problématique de l'appariement automatique de schéma d'une manière holistique (systémique) de manière à appréhender sa complexité intrinsèque et ainsi à mieux répondre aux défis existants notamment la complexité et l'incertitude ?*

Plus spécifiquement, on s'interroge sur :

- Comment peut-on aborder la complexité du processus d'appariement de telle sorte à contribuer à améliorer le résultat de l'appariement (qualité de l'alignement), et en même temps, à réduire l'effort requis pour sa configuration et son optimisation ?
- Serait-il possible d'avoir un processus d'appariement capable de s'adapter à des scénarios d'appariement différents et ce en exploitant des capacités d'auto-configuration et d'auto-optimisation ?
- Quelle serait l'orientation théorique qui pourrait représenter un cadre adéquat pour développer une nouvelle approche (holistique) pour le processus d'appariement ?

Dans cette thèse, nous visons donc, à travers une recherche à la fois empirique et théorique, à répondre à cette question de la gestion de la complexité et de l'incertitude au niveau des processus d'appariement des schémas, d'une manière holistique et systémique.

Dans la section qui suit, nous introduisons le paradigme qui est au centre de notre orientation théorique et expliquons la pertinence de son application, pour répondre à nos questions de recherche.

0.2 Approche et contributions

Les défis et difficultés engendrés par l'incertitude et la complexité qui marquent le processus de l'appariement nous ont motivés à investiguer à quel point l'application d'un paradigme émergent bio-inspiré peut nous amener à comprendre, modéliser, gérer et ultimement surmonter l'incertitude inhérente à ce processus. Ce paradigme bio-inspiré, qui a été appliqué avec succès à plein de domaines ces dernières années, est le paradigme des systèmes complexes adaptatifs (*Complex Adaptive System*) (Steels, 2000 ; Gintis, 2006 ; North *et al.*, 2013).

Un système complexe adaptatif se comporte et évolue selon deux principes clés : le premier est que l'ordre est émergent et non pas prédéterminé et le second est que l'état du système est irréversible et souvent imprévisible. Le système complexe adaptatif, de par sa nature non-linaire, fait émerger au niveau macroscopique (global) de nouvelles propriétés complexes (comportements, organisations, etc.) et ce à partir d'interactions au niveau microscopique (local) qui sont basées généralement sur des règles simples. Cela lui permet de s'auto-organiser et de s'adapter aux changements de son environnement, en l'absence d'un contrôle central gouvernant son comportement (Holland, 2006).

Les composants de base du système complexe adaptatif sont les agents (e.g. une cellule dans un système biologique, un fournisseur ou un client dans un système économique).

Les agents sont des entités qui cherchent, en évoluant dans le temps, à maximiser leur degré de satisfaction (Dooley, 1997). Ils ont des propriétés et des comportements, interagissent et s'influencent les uns les autres, apprennent de leur expérience et adaptent leur comportement à leur environnement (Holland, 2006 ; North *et al.*, 2013).

La modélisation à base d'agents « *Agent-based Modeling* » (ABM) a été utilisée avec succès, pour la modélisation des systèmes complexes adaptatifs, dans plusieurs domaines incluant la biologie, l'écologie, l'économie (Bonabeau, 2002 ; North et Macal, 2007 ; Macal et North, 2010). La simulation multi-agents[2], c'est la simulation d'un modèle à base d'agents.

Dans les systèmes complexes adaptatifs, le comportement du système n'obéit pas à une logique ou à un contrôle centralisé. La solution globale (niveau macroscopique) émerge de l'interaction des agents qui jouissent d'autonomie et qui sont donc libres de leurs comportements au niveau local et de leur adaptation à leur environnement. De la même manière, on pense que le processus d'appariement, s'il est abordé sous l'angle de système complexe adaptatif, serait capable de faire émerger la découverte des relations de correspondances entre les schémas (solution globale au niveau macroscopique) sans qu'il ait pour autant de processus centralisé qui dicte la solution finale.

L'orientation théorique de notre recherche, se base sur la théorie de la complexité et plus précisément du paradigme des systèmes complexes adaptatifs. Notre postulat de départ est que le système d'appariement de schémas peut être modélisé comme un système complexe adaptatif. Nous croyons pouvoir avancer que sous le prisme de la théorie de la complexité, l'appariement de schémas peut être vu comme : (i) un *système*, constitué de plusieurs composantes en interaction, en l'occurrence les éléments de schémas, représentés par des agents, cherchant à trouver le meilleur appariement, (ii) *complexe*, avec une dynamique non-linaire et non-déterministe, stochastique de par sa

[2] De l'anglais « *multi-agent simulation (MAS)* »

nature et donc n'est pas seulement la somme de ces constituants, (iii) *adaptatif,* c'est-à-dire dont l'autonomie de ses constituants de base (les agents) s'adapte à tous les scénarios d'appariement.

L'hypothèse centrale de cette thèse, est donc de considérer le processus de l'appariement comme un système complexe adaptatif et de le modéliser en utilisant l'approche de la modélisation et de la simulation à base d'agents. Le but visé étant l'exploitation des propriétés intrinsèques aux systèmes complexes adaptatifs (i.e. modèles à base d'agents), notamment l'émergence, la stochasticité et l'auto-organisation, pour contribuer à apporter des réponses aux questions ouvertes de l'appariement, à savoir mieux gérer la complexité et l'incertitude. Dans ce modèle à base d'agents pour l'appariement de schémas, chaque élément des schémas à apparier (schéma source ou schéma cible) est modélisé comme un agent autonome, appartenant à un groupe (groupe du schéma source ou du schéma cible) dont la simulation des comportements et de l'interaction avec son environnement, au niveau micro, fait émerger au niveau macro, un système auto-organisé[3] qui représente la solution globale à l'appariement (i.e. relations entre les éléments des schémas). En d'autres termes, la résolution du processus d'appariement passe par l'effort individuel que déploie chaque agent d'un groupe (groupe source ou cible), d'une manière autonome et locale tout au long de la simulation, pour trouver la meilleure relation de correspondance dans le groupe opposé (meilleur appariement[4]).

Notre travail de recherche se base sur cette hypothèse centrale que nous admettons dans un premier temps comme postulat. Tout au long de cette thèse nous allons tenter de démontrer sa validité (hypothèses). Ainsi, nos travaux ont pour objectif de fournir d'une part, la formalisation d'un modèle conceptuel du processus d'appariement sous

---

[3] De l'anglais « self-organized »

[4] De l'anglais « *Best Matching* »

l'angle d'un système complexe adaptatif, en utilisant l'approche de la modélisation et simulation à base d'agents. D'autre part, on veut transformer ce modèle conceptuel en un programme informatique s'exécutant sur une plateforme de simulation. Pour être plus précis, on peut résumer les principales contributions de notre approche, qui ont découlé de notre démarche de recherche, comme suit :

1. Explorer la problématique d'appariement de schémas sous l'angle d'un système complexe adaptatif :
    a. Étudier les domaines des systèmes complexes adaptatifs et de la modélisation et simulation à base d'agents
    b. Étudier des domaines connexes comme l'intelligence artificielle, incluant l'apprentissage machine, la prise de décision sous incertitude (théorie bayésienne), etc.
2. Proposer une approche holistique (contrairement aux approches réductionnistes existantes) pour le processus d'appariement de schémas dans le but de contribuer à mieux gérer la complexité et l'incertitude inhérente à ce dernier :
    a. proposer une formalisation d'un modèle conceptuel à base d'agents pour le processus d'appariement (conceptuellement agnostique à l'implémentation et aussi à l'engin de simulation);
    b. transformer ce modèle conceptuel en un programme informatique (i.e. prototype) s'exécutant sur une plateforme de simulation;
    c. concevoir un nouvel apparieur[5] basé sur l'apprentissage machine (classificateur bayésien[6]) et ce en exploitant les données générées par les simulations.

[5] De l'anglais « *matcher* »

[6] De l'anglais « *Bayesian Classifier* »

   d. Proposer une approche pour la sélection des relations de correspondance basée sur l'exécution de simulations multiples et d'une analyse statistique (quantification de l'incertitude)
3. Évaluer, d'une manière empirique, la viabilité de l'approche proposée et ce, en comparant ses performances, sur ensemble de scénarios, aux résultats attendus par l'utilisateur.

Dès lors, l'originalité de notre travail et sa pertinence viennent du fait que, au meilleur de notre connaissance, la problématique de l'appariement de schémas n'a jamais été abordée sous le prisme d'un système complexe adaptatif (i.e. modélisation de l'appariement automatique de schémas en une simulation multi-agents).

0.3 Organisation de la thèse

L'organisation en chapitres de la thèse reflète dans une large mesure le cheminement de notre travail de recherche. Nous organisons cette thèse en cinq chapitres : (i) le premier chapitre est consacré à un état de l'art sur la problématique de l'appariement automatique de schémas ainsi que sur le paradigme des systèmes complexes adaptatifs (qui est notre cadre théorique pour approcher la problématique); (ii) le deuxième chapitre est dédié à l'exposition et l'explication du cadre théorique et du cheminement méthodologique. En effet, dans ce chapitre nous expliquons comment le paradigme des systèmes complexes adaptatifs peut apporter des réponses à nos questions de recherche et comment on compte y parvenir en exploitant l'approche de modélisation et de la simulation à base d'agents. Aussi, nous proposons dans ce chapitre une formalisation d'un modèle conceptuel à base agents pour le processus d'appariement; (iii) dans le chapitre trois, on présente une première implémentation du modèle conceptuel pour l'appariement automatique des schémas basée sur des agents réflexifs; (iv) dans le chapitre quatre on procède à l'évaluation et à la validation, d'une manière empirique, de la viabilité de l'approche proposée et ce en comparant ses résultats aux résultats attendus par l'utilisateur; (v) le chapitre cinq est consacré à la proposition d'une

l'évolution de notre modèle conceptuel qui se traduit par une évolution de l'architecture interne de la composante principale de notre outil, à savoir l'agent, d'un type d'agent réflexif vers un type d'agent rationnel ou cognitif.

Enfin, nous concluons cette thèse par une synthèse de nos contributions, une analyse critique de notre approche, et des améliorations et travaux en perspective.

0.4 Publications

Notre travail de recherche, dans le cadre de cette thèse, a donné lieu à plusieurs publications (conférences internationales avec comités de lecture). Ces publications ont été non seulement bénéfiques pour la validation de notre postulat de départ, mais encore ont été d'une grande utilité pour la validation des différentes hypothèses avancées tout au long de notre cheminement de recherche. Ci-dessous, la liste de ces publications (Assoudi et Lounis, 2011, 2014c, 2014b, 2014a, 2014d, 2015a, 2015b):

- Les premiers balbutiements de l'idée de l'adaptation pour l'appariement automatique de schémas (dans le contexte d'un processus d'échange de données) :
    - Assoudi, H. et Lounis, H. (2011). Self-Healing Data Exchange Process under Evolving Schemas: A New Mapping Adaptation Approach Based on Self-Optimization.
- La proposition de l'idée de la simulation multi-agents pour l'appariement automatique de schémas (*SMAS*) (simulation basée sur des agents réflexifs)
    - Assoudi, H. et Lounis, H. (2014). Agent-based Stochastic Simulation of Schema Matching.
    - Assoudi, H. et Lounis, H. (2014). A Multi-Agent-Based Approach for Autonomic Data Exchange Processes.
    - Assoudi, H. et Lounis, H. (2014). Towards an Agent-Based Simulation Model for Schema Matching.

- Assoudi, H. et Lounis, H. (2014). Towards a Self-Organized Agent-Based Simulation Model for Schema Matching.

– La proposition d'une évolution du modèle conceptuel de l'architecture interne des agents (sur lesquels se base notre approche *SMAS*), d'un modèle d'agents réactifs vers un modèle d'agents rationnels
    - Assoudi, H. et Lounis, H. (2015). Coping with Uncertainty in Schema Matching: Bayesian Networks and Agent-Based Modeling Approach.
– Synthèse de notre travail de recherche (retour sur l'hypothèse centrale)
    - Assoudi, H. et Lounis, H. (2015b). Schema Matching as complex adaptive system.

CHAPITRE I

ÉTAT DE L'ART

1.1 Introduction

Dans la première section de ce chapitre, nous allons dans un premier temps, présenter un survol général des différentes approches et techniques existantes pour le problème de l'appariement automatique de schémas. Ensuite, dans un deuxième temps, nous allons nous atteler à présenter d'abord les principaux défis qui freinent la vision d'une automatisation complète de ce processus (e.g. l'incertitude), et nous présenterons les principales solutions proposées pour relever ces défis (e.g. optimisation). À la fin de cette section, on présentera une synthèse qui a pour objectif de faire ressortir les principales critiques que nous formulons à l'encontre des approches existantes ainsi que les limites qui, selon notre point de vue, empêchent d'apporter des réponses holistiques aux questions encore ouvertes.

La deuxième section sera dédiée la présentation du cadre théorique de notre nouvelle approche d'appariement automatique de schémas à savoir la théorie de la complexité et plus spécifiquement le paradigme des systèmes complexes adaptatifs. Dans cette section, nous allons tenter d'explorer les racines de ce paradigme et présenter ses principales caractéristiques. Aussi, nous allons exposer l'approche de modélisation d'un système complexe adaptatif à savoir la modélisation et simulation à base d'agents.

1.2 Appariement de schémas

1.2.1 Préliminaires

L'appariement de schémas est une problématique récurrente dans le domaine des bases de données et de la représentation de connaissances, comme l'intégration et l'échange des données, les entrepôts de données et l'interrogation sémantique des données (Rahm et Bernstein, 2001a). L'appariement « *Match* » est une opération fondamentale dans la manipulation des métadonnées (e.g. schémas, ontologies). Elle vise à trouver des relations de correspondances sémantiques entre les éléments des schémas de données (AH Doan *et al.*, s.d.). Dans la littérature, on discute l'appariement sous le nom d'alignement ou d'appariement.

Plusieurs définitions existent pour le processus de l'appariement de schémas. (Rahm et Bernstein, 2001a) dans leur étude sur les différentes approches pour la résolution de la problématique d'appariement de schémas, définissent un schéma comme un ensemble d'éléments reliés par une structure donnée. Le schéma doit être représenté par une notations qui permet de modéliser d'une manière naturelle et logique la notion d'élément et structure tel qu'un modèle orienté-objet, un modèle d'entité-relation, XML, ou encore sous la forme d'un modèle de graphes orientés.

S1 elements	S2 elements
Cust	Customer
C#	CustID
CName	Company
FirstName	Contact
LastName	Phone

Figure 1.1 Exemple de schémas pour l'appariement **(Rahm et Bernstein, 2001a)**

Une relation de correspondance « *Matching* » indique qu'un certain élément du schéma $S1$ est relié à un autre élément du schéma $S2$. La relation entre les deux éléments peut être décrite par une expression de transformation « *Mapping* ». L'opération de l'appariement « Match » est définit comme une fonction qui prend en entrée deux schémas $S1$ et $S2$ et retourne en sortie les correspondances entre ces deux schémas (résultat de l'appariement).

Dans sa définition de schéma, (Bernstein *et al.*, 2011) décrit le schéma comme une structure formelle représentant un artefact tel qu'un schéma SQL, un schéma XML, un diagramme d'entités-relations, la description d'une ontologie, la définition d'une interface ou bien la description d'un formulaire.

Lorsqu'il s'agit de l'appariement de schémas, il est important de souligner une distinction entre deux concepts souvent confondus, à savoir « *Schema Matching* » et « *Schema Mapping* ». Les auteurs (Bohannon *et al.*, 2006) proposent des définitions[7] pour clarifier et différencier les deux concepts. Selon eux, le concept de « *Schema Matching* » concerne la recherche des correspondances entre les éléments des schémas à apparier, alors que le concept de « *Schema Mapping* » concerne la transformation des correspondances entre les éléments des schémas à apparier.

Un état de l'art sur les différentes méthodes d'appariement que ce soit pour les schémas ou pour les ontologies a été proposé par (Shvaiko et Euzenat, 2005). De la même manière que pour les schémas, les ontologies font référence à des modèles de

---

[7] **Traduction de l'anglais :** «*Schema matching is to find a pairing of attributes (or groups of attributes) from the source schema and attributes of the target schema such that pairs are likely to be semantically related. In many systems finding such a schema matching is an early step in building a schema mapping. Schema mapping is to find a data transformation that, given an instance of a source schema, will produce an instance that conforms to a target schema while preserving the appropriate information content of the source. Finding schema mappings is a common task in a wide variety of data exchange and integration scenarios* ».

représentation de connaissances. Les ontologies, par contre, se démarquent principalement par rapport aux schémas, par l'apport d'une manière formelle et explicite de la sémantique au niveau des données. De plus, les auteurs proposent des définitions des principaux termes de la problématique de l'appariement de schémas, à savoir les termes « *appariement* », « *alignement* », ainsi que le terme « *élément d'alignement* ». Les définitions proposées sont les suivantes :

- Un *élément d'appariement* « *mapping element* » se définit par un quintuplet $\langle id, e, e', n, R \rangle$ où id est un identifiant pour le lien; e et e' sont les entités mises en relation ; n est une mesure de confiance du lien et R est la relation qui relie e et e' (e.g. équivalence $(=)$; plus générale $(\sqsupseteq)$; disjonction $(\perp)$; overlapping $(\sqcap)$)
- *Un alignement* « *alignment* » est un ensemble de liens d'appariement.
- *L'appariement* « *matching* » est le processus qui permet de générer un alignement (A') pour une paire de schémas/ontologies (O, O'). Il existe d'autres paramètres qui peuvent étendre la définition du processus d'appariement, à savoir: (i) l'utilisation d'un alignement en entrée (A) qui doit être complété par le processus; (ii) les paramètres pour l'appariement, p (e.g. poids, seuils); et (iii) les ressources externes utilisées par le processus d'appariement, r (e.g. thesaurus); voir la figure 1.2.

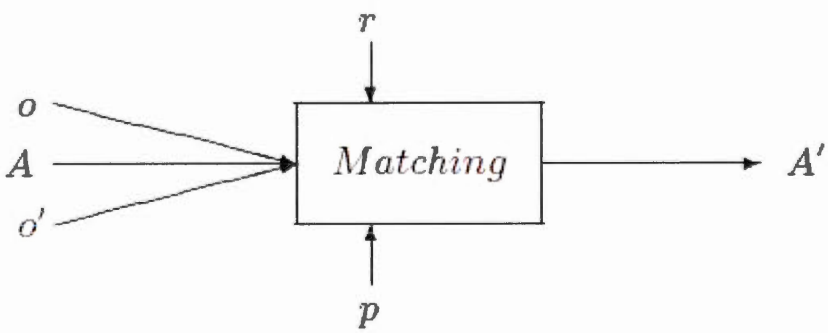


Figure 1.2 Processus de l'appariement **(Shvaiko et Euzenat, 2005)**

Le processus d'appariement de schémas, est souvent utilisé comme étape préalable à la résolution d'autres problématiques, telles que l'intégration ou l'échange des données ou encore l'évolution des schémas (Fagin *et al.*, 2009). En effet, dans le cadre d'un processus d'intégration ou d'échange des données[8] (Arenas et Libkin, 2008 ; Fagin *et al.*, 2005 ; Miller *et al.*, 2001 ; Tian *et al.*, 2013), le processus d'appariement devient la tâche qui est responsable de trouver un alignement qui soit équivalent sémantiquement entre un schéma source et un schéma destination. Cet alignement, dans le cas de l'intégration des données par exemple, va participer à trouver des « réponses aux requêtes[9] » formulées à plusieurs sources de données disparates et ce en consolidant les schémas de ces sources disparates vers un schéma commun (c.à.d. un schéma global[10]). Dans le cas de l'échange des données, trouver un alignement entre un schéma source et un schéma cible, sert à l'échange des données entre des systèmes ou des applications d'entreprise hétérogènes (tels que les systèmes *ERP*, les bases de données, ou les systèmes patrimoniaux) et ce en aidant à transformer les données du format du système source vers le format du système cible.

Quant à la problématique de l'évolution de schémas[11] (McCann *et al.*, 2005 ; Velegrakis *et al.*, 2003, 2004), le processus d'appariement contribue à la correction d'un alignement existant qui a été impacté par le changement du schéma source et/ou schéma cible (adaptation de l'alignement[12]). En effet, il s'agit d'utiliser le processus d'appariement pour trouver un alignement qui soit équivalent sémantiquement au précédent alignement qui a été impacté par le changement.

---

[8] De l'anglais « *Data Integration and Data Exchange*»

[9] De l'anglais (*Query Answering*)

[10] De l'anglais (*Global or Mediated Schema*)

[11] De l'anglais « *Schema evolution* »

[12] De l'anglais « *Mapping Adaptation* »

Comme déjà mentionné, une myriade d'approches et de techniques a été proposée dans la littérature pour la problématique de l'appariement automatique de schémas. Dans leur revue sur l'appariement de schémas, Bernstein et al. (Bernstein *et al.*, 2011), ont proposé un recensement des principales approches. Ils ont regroupé ces approches en deux catégories : (i) d'un côté, les techniques classiques qui avaient déjà été discutées en 2001 (Rahm et Bernstein, 2001a), (ii) et de l'autre côté, les techniques récentes (proposées depuis 2001). Parmi les techniques classiques, on trouve :

- L'appariement basé sur la linguistique : basé sur la similarité entre les attributs des éléments de schémas, comme le nom ou la description. La similarité est calculée en exploitant des techniques d'analyse de texte (e.g. tokénisation, lemmatisation) et d'extraction de l'information.
- L'appariement basé sur une source auxiliaire d'information : exploite des thésaurus, dictionnaires (e.g. acronymes, synonymes) externes pour trouver des correspondances.
- L'appariement basé sur les instances : se base sur la similarité des instances des éléments de schémas et ce en utilisant des statistiques ou des classificateurs d'apprentissage machine.
- L'appariement basé sur la structure des schémas : se base sur la similarité trouvée à partir de la structure des groupes auxquels appartiennent les éléments de schémas (relations similaires entre éléments).
- L'appariement basé sur les contraintes : exploite les contraintes définies au niveau des éléments de schémas (e.g. le type, les contraintes d'intégrité) pour trouver des correspondances.
- L'appariement basé sur les règles : se base sur des règles d'appariement exprimées en logique de premier ordre pour trouver des correspondances ;
- L'appariement hybride : combine plusieurs techniques.

Toujours selon la revue des techniques existantes (Bernstein *et al.*, 2011), depuis 2001, plusieurs techniques récentes, utilisant des algorithmes exploitant d'autres types d'information, ont vu le jour :

- L'appariement basé sur les graphes : basé sur la comparaison des relations entre les éléments des graphes (représentant les schémas). Cette comparaison se base, par exemple, sur la propagation des similarités en fonction des arcs des graphes (c.à.d. comparaison de la similarité entre les nœuds adjacents) (Melnik *et al.*, 2002) ou encore par l'utilisation d'algorithmes de satisfaction de contraintes probabilistes (AnHai Doan *et al.*, 2002)
- L'appariement basé sur le contenu des documents : les instantes d'un élément de schéma sont groupés dans un document qui est ensuite comparé à un autre document (représentant les instantes d'un autre élément), en utilisant des mesures de similarité *TF-IDF* (utilisée dans le domaine de l'extraction de l'information) (Li *et al.*, 2009 ; Massmann et Rahm, 2008)
- L'appariement basé sur l'utilisation : à titre d'exemple, l'idée qui consiste à exploiter les informations extraites des journaux de requêtes pour trouver des correspondances entre les éléments des schémas à apparier (Elmeleegy *et al.*, 2008).

La figure 1.3 (Rahm et Bernstein, 2001a) présente une classification des différentes stratégies proposées pour le résolution de la problématique de l'appariement de schémas. À très haut-niveau, on trouve deux classes d'approches, d'un côté celles basées sur un apparieur individuel et de l'autre, celles basées sur une combinaison d'apparieurs.

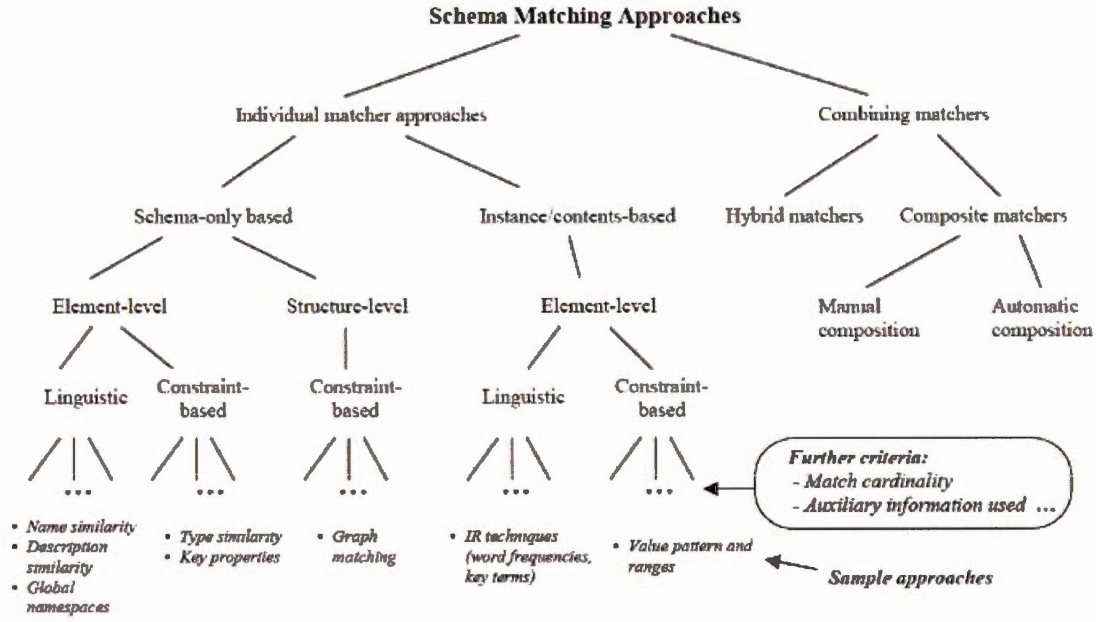


Figure 1.3 Classification des techniques classiques **(Rahm et Bernstein, 2001a)**

Ces différentes approches ont donné lieu à une foule d'outils qui diffèrent, notamment, en fonction de l'importance accordée au défi à résoudre et au mode d'utilisation (e.g. appariement automatique ou semi-automatique, processus configurable et adaptable). En effet, l'appariement automatique de schémas est un sujet de recherche qui demeure très actif, comme en témoigne les nombreux prototypes et outils qui ont été proposés dans la littérature. Parmi les outils, issus de la recherche, les plus populaires sont :

– *CUPID* (Madhavan *et al.*, 2001): cet outil d'appariement est basé sur une approche hybride qui combine à la fois une approche de similarité basée sur les éléments et une approche de la similarité basée sur les structures (i.e. le degré de similarité est une combinaison d'une similarité linguistique et une similarité structurelle). La première phase de l'appariement calcule la similarité linguistique correspondant à une similarité calculée au niveau des noms d'éléments. Durant la deuxième phase, les schémas en entrée sont transformés

sous la forme de graphes et ensuite le degré de similarité est calculé selon des règles prédéfinies concernant l'emplacement de l'élément dans la structure du schéma (e.g. parents, enfants). La sélection de l'alignement est ensuite effectuée en choisissant les paires ayant un degré de similarité pondéré maximal.

- *LSD* (*Learning Source Descriptions*) (AnHai Doan *et al.*, 2000, 2001): ce système utilise une technique d'apprentissage machine (apprenants[13]) pour produire un alignement d'une cardinalité 1:1. Cet alignement est le résultat d'une combinaison, par le biais d'un méta-apprenant, des prédictions de plusieurs apprenants de base (i.e. apprenant dédié pour les noms des éléments de schémas ou encore apprenants dédiés pour l'apprentissage à partir des instances de données).
- *Similarity Flooding* (Melnik *et al.*, 2002): cet outil se base sur l'idée de l'appariement de graphes. Il commence par la conversion des modèles à apparier (diverses structures de données, schéma de données) en graphes étiquetés orientés[14], ensuite calcule la similarité entre les nœuds de ces graphes en se basant sur l'idée que les éléments de deux modèles distincts sont similaires quand leurs éléments adjacents sont aussi similaires. En d'autre terme, une partie de la similarité de deux éléments se propage vers leurs éléments adjacents respectifs (voisins), d'où le nom de l'outil : *Similarity Flooding.*
- *Clio* (Miller *et al.*, 2001) : c'est un outil semi-automatique développé par *IBM*[15]. Il se base sur une approche hybride combinant la mesure de similarité textuelle

[13] De l'anglais "*Learners*"

[14] De l'anglais « Directed Labeled Graphs »

[15] CLIO Project developpé conjointement entre *IBM Almaden Research Center* et *Department of Computer Science, Univesity of Toronto.* CLIO est l'un des rare outil à avoir passé d'un prototype de

pour le nom des éléments de schéma avec un apparieur exploitant les techniques de l'apprentissage machine appliquées aux données des instances des schémas. Cet outil dispose d'une interface utilisateur permettant un appariement interactif.

- *COMA* (Do et Rahm, 2002) : cet outil d'appariement est un outil hautement configurable. Il permet la création de différentes stratégies d'appariement et ce en permettant la combinaison d'un ensemble important d'apparieurs et de paramètres d'appariement.
- *iMap* (Dhamankar *et al.*, 2004) : c'est un outil d'appariement semi-automatique permettant la découverte des appariements de cardinalité 1:1 ainsi que les appariements complexes. L'outil iMAp se compose de trois modules principaux : un générateur de correspondances candidates, un estimateur produisant une matrice de similarité avec des paires entre élément candidat et élément cible et finalement, un sélecteur qui à partir de la matrice de similarité sélectionne le meilleur appariement.
- *S-Match* (Giunchiglia *et al.*, 2004) : il s'agit d'un outil d'appariement sémantique entre deux graphes (e.g. schémas ou ontologies en entrée). Il calcul la plus forte relation sémantique entre les libellés des différentes paires de nœuds des deux graphes.

Malgré la multiplication des approches, des techniques et des outils d'appariement, il n'en demeure pas moins, qu'à haut niveau, elles partagent toutes des similitudes profondes au niveau du fonctionnement du processus d'appariement.

---

recherche à un outil commercial (Clio fait partie de l'outil IBM Rational Data Architect) (Haas et al., 2005)

1.2.2 Processus d'appariement

En ce qui a trait au fonctionnement du processus d'appariement automatique de schémas, il se déroule, d'une manière générale, en deux étapes (Gal, 2006a). Lors de la première étape, le degré de similarité (score) entre toutes les paires d'éléments des deux schémas est calculé et une matrice de similarité est produite. Comme deuxième étape, et à partir de la matrice de similarité, une seule correspondance est identifiée et sélectionnée comme la *meilleure correspondance*[16]. Cette sélection de la meilleure correspondance est basée généralement sur une fonction qui doit satisfaire certaines contraintes (e.g. un degré minimal de similarité[17]).

D'une manière plus détaillée, comme le montre la figure 1.4, le processus d'appariement peut être vu comme une série d'étapes (*flux de travail*[18]) commençant par l'importation des schémas en entrées, et se terminant par la production d'un alignement en sortie, en passant successivement par l'étape de prétraitement (e.g. transformation des schémas en graphe), de l'étape de l'exécution de l'apparieur (i.e. l'exécution d'un sous-processus impliquant l'utilisation de différentes techniques et stratégies), suivie de l'étape de la combinaison des résultats produits par les différents appariеurs et enfin de l'étape de la sélection de l'alignement à partir des résultats combinés (Rahm, 2011).

---

[16] De l'anglais « *Best Match* »

[17] De l'anglais « *minimal degree of similarity*»

[18] De l'anglais « Workflow »

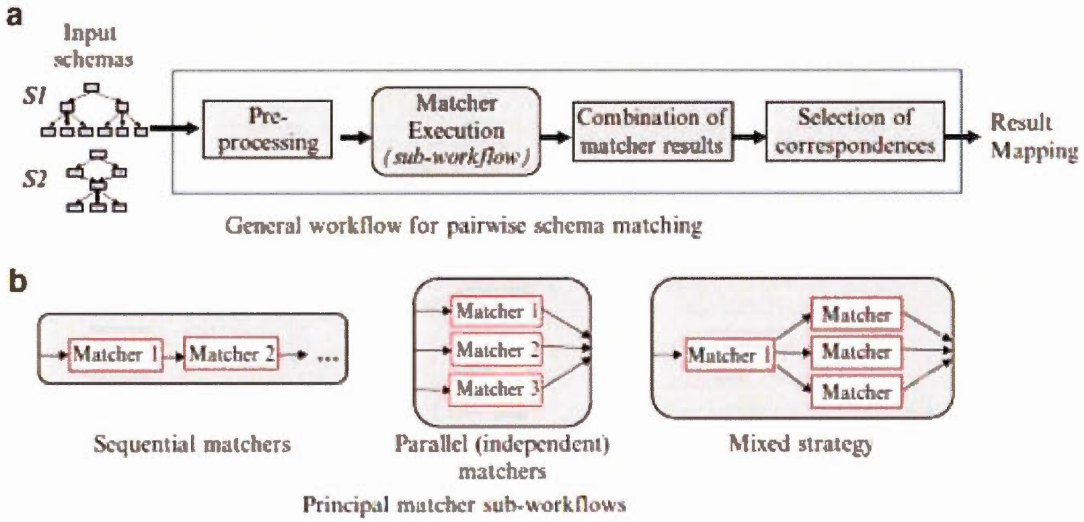


Figure 1.4 Processus général d'appariement de schémas **(Rahm, 2011)**

Outre la représentation des étapes du processus d'appariement, à savoir le calcul de similarité, l'agrégation des résultats et la sélection de l'alignement, comme un flux de travail, le processus d'appariement peut aussi être représenté sous la forme d'un graphe dirigé acyclique décrivant l'ordre d'exécution des opérations d'appariement (Peukert *et al.*, 2012). Les opérations dans ce graphe peuvent recevoir une ou plusieurs matrices de similarités en entrée, et retourne une matrice de similarités en sortie. L'assemblage de ces opérations produit différentes topologies de base pour le processus d'appariement. Ces topologies peuvent varier. En revanche, les topologies les plus utilisées restent la topologie parallèle, séquentielle ou itérative. Ces topologies de base peuvent être combinées avec d'autres stratégies d'appariement plus sophistiquées.

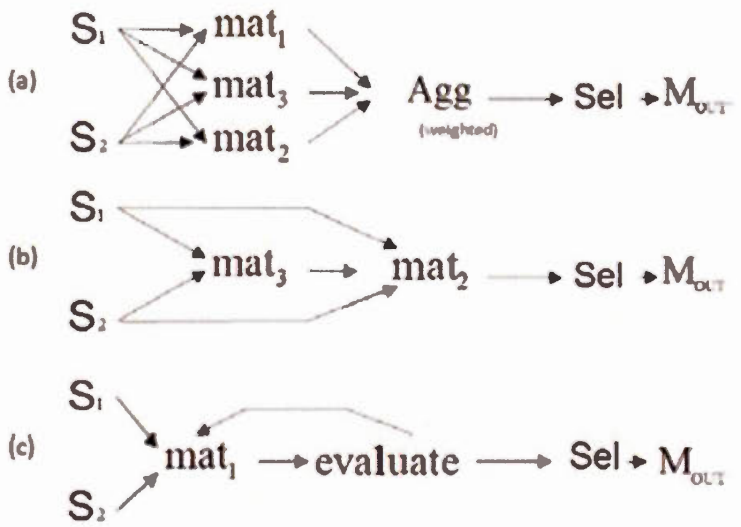


Figure 1.5 Topologies des processus d'appariement **(Peukert *et al.*, 2012)**

Le concept de processus d'appariement s'avère donc un concept assez flexible pouvant être pensé et représenté de différentes manières (e.g. sous la forme d'un flux de travail ou sous la forme d'un graphe dirigé acyclique). Mais, quel que soit la forme que peut prendre le processus d'appariement, il est clair que les étapes constituant le noyau dur restent quasiment les mêmes (avec des variations mineures, selon les approches, au niveau de l'étape de pré-traitement ou celle du post-traitement) à savoir : (i) l'étape de la sélection et de la combinaison des appariеurs, (ii) l'étape de la combinaison des résultats des appariеurs, (iii) et enfin celle de la sélection de l'alignement.

1.2.2.1 Sélection et exécution des appariеurs

Cette étape du processus d'appariement se compose de deux tâches distinctes mais complémentaires. La première concerne la sélection des appariеurs et la seconde concerne l'exécution des appariеurs sélectionnés.

En ce qui trait à la sélection des appariеurs, cette tâche peut être accomplie manuellement par l'utilisateur ou automatiquement par l'outil lui-même. Dans le cas où c'est une sélection manuelle, l'utilisateur en se basant sur son expérience, lors de la

phase de conception (logique fixe dans le code) ou lors de la phase de configuration (pour les approches qui offrent la possibilité de configuration selon les scénarios d'appariement), sélectionne les appariеurs à utiliser par le processus d'appariement. À titre d'exemple, on trouve l'outil *COMA* (Do et Rahm, 2002) qui à l'aide de fichiers de configuration permet à l'utilisateur de créer des stratégies d'appariement incluant le choix des appariеurs. En revanche lorsqu'il s'agit d'une sélection automatique, l'outil accomplie automatiquement la sélection des appariеurs en se basant sur des techniques tel que l'apprentissage machine. Un bon exemple d'outils permettant la sélection automatique serait *YAM* (Duchateau *et al.*, 2009) qui propose une approche basée sur l'apprentissage machine, pour la sélection automatique des appariеurs adaptés au scénario d'appariement envisagé. En effet, les auteurs de *YAM* définissent leur approche comme un générateur d'appariеurs à la carte (fabrique d'appariеurs). Il serait pertinent, ici, de noter que la majorité des outils d'appariement propose une approche manuelle pour la sélection des appariеurs.

La tâche d'exécution des appariеurs quant à elle, a pour objectif de définir les stratégies régissant l'orchestration de l'exécution des différents appariеurs. En effet, et comme le montre la figure 1.4, les appariеurs individuels peuvent soit être exécutés séquentiellement, en parallèle (indépendamment), ou dans un mode mixte. Dans le mode séquentiel, les appariеurs individuels sont exécutés de telle sorte que l'extrant (résultat) d'un appariеur devient l'intrant d'un autre appariеur. L'outil *Cupid* (Madhavan *et al.*, 2001), par exemple, a pour approche d'exécution d'appariеur, de d'abord exécuter un appariеur linguistique (permettant de comparer les noms des éléments de schéma) et ensuite utiliser les scores de similarité obtenus comme entrée pour un appariеur structurel (permettant de comparer la structure des schémas). Concernant le mode parallèle, les appariеurs individuels sont autonomes et peuvent être exécutés indépendamment les uns des autres. Cela permet une grande flexibilité pour sélectionner les appariеurs pour l'exécution et la combinaison. En outre, ces appariеurs peuvent aussi être physiquement exécutés en parallèle, sur par exemple, des

microprocesseurs multi-cœur. Enfin, le mode mixte combine l'exécution séquentielle des appariеurs ainsi que l'exécution en parallèle, ce qui confère à ce mode une plus grande complexité.

1.2.2.2 Combinaison des résultats

L'exécution d'un appariеur produit en général une matrice de similarités (Gal, 2011a). La matrice de similarités, celle-là même qui se trouve au cœur du processus d'appariement, est une matrice représentant le degré de similarité entre un élément de schéma source et un élément du schéma cible. Le degré de similarité est généralement un nombre situé à l'intérieur d'un intervalle entre 0 et 1.

Comme on l'a vu plus haut, le choix du mode parallèle, pour l'exécution des appariеurs, permet aux appariеurs de s'exécuter indépendamment les uns des autres et de produire ainsi plusieurs matrices de similarités. Ces dernières sont ensuite groupées sous la forme d'un cube de similarités (agrégation de plusieurs matrices). À la suite de quoi, une opération de combinaison de similarités est exécutée et qui a pour but de réduire le cube vers une simple matrice de similarités qui contient les meilleurs scores de similarité. L'opération qui permet de réduire le cube de similarités en une matrice de similarités (combinaison des résultats) est appelée la combinaison des similarités (Peukert *et al.*, 2010).

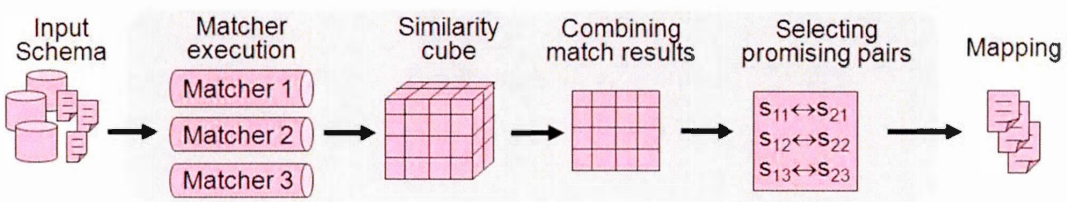


Figure 1.6 Processus d'appariement **(Peukert *et al.*, 2010)**

L'opération qui consiste à combiner les résultats de calcul de similarités des différents appariеurs (réduction d'un cube de similarités vers une matrice de similarités) est souvent basée sur des fonctions d'agrégation tels que : min, max, moyenne ou encore une moyenne pondérée[19] (Bellahsene et Duchateau, 2011).

Un exemple d'agrégation est l'agrégation d'apparieur terminologique (basée sur des mesures de similarité textuelles) avec le résultat d'un apparieur structurel (basé sur des mesures de similarité entre structures). Le poids donné à chaque apparieur devient un facteur influençant le résultat final. C'est pour cette raison que l'optimisation du processus d'appariement implique souvent l'optimisation de ce paramètre, à savoir le poids de chaque apparieur lors de l'agrégation des résultats. Ce paramètre est généralement défini en se basant sur l'expérience de l'utilisateur, ou dans le cas de certains outils (cas rares), en se basant sur un processus automatique d'optimisation.

Souvent, des techniques d'apprentissage machine sont utilisées pour l'automatisation de la sélection d'une valeur optimale du paramètre poids. Cependant, l'utilisation de l'apprentissage machine peut poser certains problèmes. En effet, l'apprentissage supervisé requiert la disponibilité d'un ensemble d'exemples étiquetés (i.e. ensemble de correspondances correctes fournies par l'utilisateur), qui ne sont pas toujours disponibles. D'un autre côté, l'hétérogénéité et la diversité qui peut caractérisé les schémas ayant servi à entrainer le *classificateur* et les schémas à apparier peut conduire à avoir une valeur inadéquate pour le paramètre poids (Peukert, 2013).

L'étape qui vient après la combinaison des résultats est l'étape de la sélection des pairs les plus prometteuses, en discriminant les autres grâce à différentes techniques (e.g. seuils).

---

[19] Normes triangulaires « *T-Norm* »

1.2.2.3 Sélection de l'alignement

Pour l'étape de la sélection de l'alignement, le plus souvent, deux types de stratégies peuvent être adoptées (individuellement ou d'une manière combinée) : (i) d'un côté les stratégies basées sur les seuils[20] , et de l'autre, (ii) les stratégies basées sur le maximum[21].

Le principal avantage des stratégies basées sur les seuils est leur simplicité. Par contre l'utilisation d'un seuil fixe présente un défi quant à la détermination de la valeur optimale qui permet d'accroitre la précision, sans pour autant discriminer des correspondances valides. Autrement dit, si le seuil est trop haut la sélection devient restrictive par contre si le seuil est trop bas, la sélection devient trop permissive (Melnik *et al.*, 2002).

Concernant les stratégies basées sur le maximum, on trouve *MAX-N* et *MAX-DELTA* (Do et Rahm, 2002). La stratégie *MAX-N* va sélectionner un nombre fixe *N*, des meilleurs candidats pour être un élément d'alignement. La stratégie *MAX-DELTA* quant à elle ne se base pas sur un nombre fixe *N*, mais se base plutôt sur un delta. En plus de la correspondance avec le score de similarité maximal, qui est retenu comme correspondance candidate, les autres correspondances ayant un score de similarité qui se situe dans un intervalle déterminé par le score maximal et la valeur delta. Cette valeur delta va alors influencer le résultat de sélection (une valeur delta très faible peut amener à sélectionner seulement la correspondance avec le meilleur score à savoir *Max-1*). Même si la définition de la valeur delta devient alors (comme on l'a vue pour

[20] De l'anglais « Threshold-based Strategies »

[21] De l'anglais « Maximum-based Strategies »

le seuil) un exercice qui influence le résultat de sélection, la stratégie de *MAX-DELTA* est considérée comme robuste (Aumueller *et al.*, 2005 ; Peukert, 2013).

Il serait pertinent de noter ici que la sélection des meilleures correspondances peut se faire du point de vue de l'élément du schéma source ou de l'élément du schéma destination ou les deux. Cette précision nous amène à la notion de direction, qui a elle aussi un impact sur le résultat de la sélection.

Les techniques de sélection peuvent être combinées et appliquées séquentiellement. En effet, on trouve souvent que la technique du seuil est utilisée comme une première étape de sélection permettant d'éliminer les correspondances peu probables avant d'enchaîner avec une autre technique (e.g. *MAX-DELTA*) pour la sélection des correspondances candidates (Do et Rahm, 2002).

Les stratégies les plus robustes sont la moyenne pour la combinaison et la stratégie *MAX-DELTA* pour la sélection. En revanche, il n'existe pas de technique de combinaison et de sélection qui peut être performante pour tous les scénarios (Peukert, 2013).

1.2.2.4 Synthèse

Le tableau 1.1, recense les différentes approches discutées plus haut (liste non exhaustive), et utilisées lors des étapes les plus importantes du processus d'appariement.

Étape	Description	Stratégies
Sélection et exécution des appariers	Sélectionner différents appariers (e.g. mesures de similarité) pour le calcul de similarités	La sélection des appariers peut être soit : - Une sélection manuelle des appariers (i.e. logique dans le code ou fichier de configuration) - Une sélection automatique des appariers (i.e. heuristiques, règles, technique d'apprentissage automatique)
	Définir les stratégies régissant l'orchestration de l'exécution des différents appariers.	les appariers individuels peuvent soit être exécutés en mode séquentiel, en mode parallèle (indépendamment) ou encore en mode mixte
Combinaison des résultats	réduire le cube vers une simple matrice de similarités qui contient les meilleurs scores de similarité	la combinaison des similarités peut être basée sur des fonctions d'agrégation tel que : min, max, moyenne ou encore une moyenne pondérée
Sélection de l'alignement	sélection des correspondances les plus prometteuses	Les stratégies de sélection peuvent être : - Des stratégies basées sur des seuils ou basées sur le maximum (i.e. *MAX-N et MAX-DELTA*) - Des stratégies appliquées selon différentes directions telles que : unidirectionnel, bidirectionnel, ou encore selon la technique « *Marriage Problem* »

Tableau 1.1 Principales étapes du processus de l'appariement de schémas

1.2.3 Mesures de similarité

La pierre angulaire de toutes les approches d'appariement automatique de schémas, est sans nul doute, le calcul de la similarité entre les éléments. Ce calcul repose généralement sur des mesures de similarité. Deux grande catégories de mesures sont exploitées pour l'appariement automatique : (i) mesures textuelles (lexicales et sémantiques) et (ii) mesures structurelles (similarité entre graphs). Concernant les

mesures de similarité textuelle, une pléiade de mesure de similarités existe. Malgré leurs différences, elles ont toutes pour objectif de répondre à la question concernant le degré (score) de similarité pour chaque paire d'éléments de schémas et ce en calculant la similarité de leurs métadonnées (nom, description, etc.). Ce calcul de similarité est une étape incontournable dans le processus d'appariement de schémas.

Les mesures de similarité textuelles peuvent êtres classifiées en trois grandes catégories (W. Cohen *et al.*, 2003):

- Des mesures de distance d'édition pour les caractères : dans cette catégorie on retrouve des mesures se basant sur des fonctions de distance qui considèrent les chaînes de caractères à comparer comme une séquence de caractères dont la similarité est déterminée par des caractères communs et la position de ces caractères dans les chaînes (Winkler, 1990) ou par le nombre d'opérations (suppression; insertion; remplacement) nécessaires pour construire une chaîne à partir de l'autre (Levenshtein, 1966). Une autre fonction connue, qui est une variante plus optimisée de la fonction Smith-Waterman, est la fonction de distance Monge-Elkan (Monge et Elkan, 1997).
- Des mesures de distance pour les mots : ces méthodes, basées sur la distance entre les mots, considèrent les deux chaines à comparer comme un ensemble de mots « *bag of words* » (c.-à-d. sous-chaines de caractères délimitées par des caractères spécifiques tel qu'un espace). Parmi les mesures les plus populaires dans cette catégorie, on retrouve :
    - la mesure de *Jaccard* (Jaccard, 1901), qui détermine le rapport entre le nombre de mots communs « *overlap* » et le nombre total de mots distincts. Cette mesure calcul $\frac{S \cap T}{S \cup T}$ pour deux ensembles de mots $\{S\}, \{T\}$.
    - La métrique *TF-IDF* « *term frequency-inverse document frequency* », qui est largement utilisée dans le domaine de l'extraction de

l'information, et dont l'idée principale est d'accorder, dans l'évaluation de la similarité entre deux chaînes de caractères, une plus grande importance aux mots peu communs. Rapport entre la fréquence du mot ou terme dans la chaine de caractères (i.e document) et l'inverse de sa fréquence dans les deux chaines de caractères (i.e. ensemble des documents ou corpus) à comparer.

  - La mesure de similarité *Cosine*, quant à elle, détermine à quel point deux chaines de caractères sont similaires en calculant l'angle formé entre deux vecteurs dans un espace multidimensionnel. Plus l'angle entre les vecteurs est faible, et plus ceux-ci sont similaires.

- Des mesures hybrides : Ces approches hybrides décomposent, dans un premier temps, une paire de chaînes de caractères en lexèmes et dans un deuxième temps, comparent les lexèmes à l'aide d'une fonction de similarité basée sur l'analyse des caractères. A titre d'exemple, la mesure *SoftTFIDF* (W. Cohen *et al.*, 2003) qui a été proposée comme une version « souple » de la méthode *TF-IDF* et qui a pour objectif de tenir compte, non seulement des lexèmes communs aux deux chaînes de caractères comparées, mais aussi des lexèmes similaires.

Les mesures de similarité sémantiques quant à elles permettent de calculer le degré de similarité sémantique et ce en utilisant des sources auxiliaires externes tel que *WordNet*, *Wiktionary* et *Wikipedia*.

Certains travaux ont proposé des cadriciels[22] implémentant un ensemble de mesures de similarité sous la forme de bibliothèques :

- *SecondString* (W. W. Cohen *et al.*, 2003) : une librairie implémentée en java intégrant plusieurs mesures de similarité textuelle connues.
- *SimMetrics* (Chapman, 2005) : une librairie java contenant exclusivement des mesures de similarité textuelles. Ces mesures calculent des similarités lexicales en comparant des séquences de chaines de caractères (sans aucun traitement sémantique). Parmi les mesures comprise dans cette librairie on trouve la mesure (Levenshtein, 1966) ou encore la mesure (Monge et Elkan, 1997).
- *DkPro* (Bär *et al.*, 2013) : c'est un cadriciel à vocation à réunir une grandes variétés de mesures de similarité dans un même et unique référentiel de mesures (libraire java) et ce en les implémentant avec interfaces standards (exploite entre autres les mesures des deux librairies précédemment citées *SecondString* et *SimMetrics*). À titre d'exemple des mesures textuelles que contient la librairie de ce cadriciel, on trouve des mesures telles que la mesure de *Jaro-Winkle*r (Winkler, 1990) ou encore la mesure *Greedy String Tiling* (Wise, 1996). Outre les mesures textuelles, *DkPro* implémente plusieurs mesures sémantiques, structurelles, stylistiques ou encore phonétiques. En plus des mesures de similarité, ce cadriciel permet d'inclure différents composants (e.g. composants pour la lecture des ensembles de données, composants de l'apprentissage machine) permettant l'intégration avec n'importe quel pipeline *UIMA*[23].

[22] De l'anglais (*Frameworks*)

[23] UIMA est un Standard Apache. « Unstructured Information Management applications are software systems that analyze large volumes of unstructured information in order to discover knowledge that is relevant to an end user. UIMA is a framework and SDK for developing such applications." Source www.apache.org.

1.2.4 Défis et difficultés

Outre la complexité qui découle du haut degré d'hétérogénéité, caractérisant généralement les schémas à apparier, et qui rend la tâche de l'appariement de ces schémas une tâche intrinsèquement complexe (Rahm, 2011), plusieurs travaux de recherche se sont penchés sur la résolution d'autres problèmes identifiés comme complexes (Bellahsene Bonifati et Rahm, 2011). Ces derniers peuvent être déclinés en des défis majeurs comme suit :

- Incertitude, optimisation et complexité du processus d'appariement : ce problème concerne l'incertitude, due à l'hétérogénéité, qui caractérise le résultat du processus d'appariement, ainsi que l'exercice qui vise à réduire cette incertitude, en l'occurrence l'optimisation.
- L'appariement des schémas à grande échelle : ce défi concerne la performance lorsqu'il s'agit de l'appariement d'un grand nombre de schémas et/ou l'appariement de schémas de grandes tailles (Do et Rahm, 2007).
- L'appariement en temps réel : ce défi concerne les cas qui nécessitent un appariement en temps réel pour des environnements hautement dynamiques (e.g. sites web composites[24] ou agents communication).
- L'alignement complexe : ce défi concerne l'appariement qui consiste à trouver des éléments d'alignement complexes (correspondance complexe) appariant un ou plusieurs éléments du schéma source avec un ou plusieurs éléments du schéma cible (c.-à-d. une correspondance de cardinalité n:m au lieu d'une cardinalité 1:1) (Aumueller *et al.*, 2005).

---

[24] De l'anglais "*Mashup Websites*"

Après le défi de l'incertitude, le défi de l'appariement des schémas à grande échelle (c.à.d. problème de montée en charge[25]) est un autre défi qui est souvent abordé dans la littérature. L'auteur (Rahm, 2011), avance que malgré les avancées faites par les approches d'appariement actuelles, elles ont toujours du mal à répondre à deux défis majeurs, surtout pour l'appariement de schémas à grande échelle, qui sont l'efficacité (e.g. l'optimisation de la qualité du résultat) et l'efficience (e.g. l'optimisation du temps de réponse). Plusieurs travaux se sont penchés sur la question de l'optimisation du temps de réponse, afin de proposer des approches permettant d'apporter une réponse à ce problème d'appariement à large échelle. Parmi ces approches, on peut citer : *COMA* (Do et Rahm, 2002), *Falcon-Ao* (Hu et Qu, 2008), *MOM* (*Modularization based approach*) (Wang *et al.*, 2006) et *Malasco* (*Matching large scale ontologies*) (Paulheim, 2008). Ces approches, pour la plupart, abordent le problème de l'appariement à large échelle sous l'angle de la décomposition en sous problèmes. En effet, ces approches d'appariement de schémas ou d'ontologies suivent l'approche « diviser pour régner » (*divide-and-conquer*) pour, dans un premier temps, partitionner les larges schémas ou ontologies en entrée en de plus petits blocs (ou grappes) et, dans un deuxième temps, pour trouver les alignements au niveau de ces blocs.

Cette thèse se concentre, d'un côté, sur le défi de l'incertitude engendrée par l'hétérogénéité, et de l'autre côté, sur le défi de l'optimisation visant la réduction de cette incertitude ainsi que la complexité qui découle inévitablement de cet exercice.

1.2.4.1 Hétérogénéité et incertitude

Le défi de l'incertitude, inhérente au résultat du processus d'appariement, est principalement dû à l'ambiguïté et l'hétérogénéité intrinsèques aux schémas. En effet, les schémas à apparier sont souvent de nature hétérogène, que ça soit au niveau de la description de leurs éléments (e.g. libellés, commentaire), au niveau de leurs structures

[25] Traduction de l'anglais (*Scalability issue*)

ou encore au niveau de leurs instances (i.e. données). La réduction de cette incertitude passe généralement par un exercice d'optimisation du processus de l'appariement qui va avoir pour objectif d'améliorer la qualité des résultats avec idéalement un effort limité. La gestion, la modélisation et ultimement la réduction de l'incertitude dans le processus d'appariement des schémas reste une question de recherche ouverte qui a motivé plusieurs travaux de recherche (Madhavan *et al.*, 2002 ; Gal *et al.*, 2005 ; Gal, 2006b, 2006a ; Nottelmann et Straccia, 2007 ; Castano *et al.*, 2008 ; Gal, 2011b ; Gong *et al.*, 2012 ; Nian-Feng et Xing-Chun, 2012 ; Rizopoulos, 2010 ; Zhang *et al.*, 2013).

L'incertitude, dans l'appariement de schémas, est considérée d'une part comme une résultante inévitable que l'on peut accepter ou même chercher à exploiter, et d'autre part elle est considérée comme intolérable et par le fait même devant être réduite au maximum.

En effet, en ce qui concerne l'idée qui consiste à accepter l'incertitude, on soutient (Bernstein *et al.*, 2011) que même les meilleurs algorithmes d'appariement de schémas font beaucoup d'erreurs, en particulier quand il s'agit d'algorithmes entièrement automatiques où l'humain n'est pas impliqué. Malgré ces erreurs, certaines applications peuvent utiliser les résultats du processus d'appariement tel quel. Cela est particulièrement le cas lorsque l'appariement de type « *meilleur-effort*[26] » est satisfaisant ou lorsque le résultat du processus d'appariement n'est pas une finalité en soit (contribuant à aider l'utilisateur final dans la résolution d'une autre tâche). Au-delà de l'idée d'accepter et de « cohabiter » avec l'incertitude, vient l'idée d'exploiter. On soutient qu'une des façons de confronter le problème de l'incertitude, générée par l'appariement, c'est de la représenter d'une manière explicite et de la considérer comme une résultante importante de ce processus (Magnani et Montesi, 2010). Cette

[26] De l'anglais « *best-effort* »

résultante peut alors être considérée comme une source importante d'information et non pas comme un problème à éviter.

À l'opposé de ce qui précède, on trouve l'idée de la maitrise et de la réduction de l'incertitude. En effet, plusieurs solutions ont été proposées dans la littérature, pour modéliser l'incertitude. Ces solutions se divisent en deux grandes catégories : (i) approches intrinsèques, et (ii) approches extrinsèques.

Concernant les approches intrinsèques, elles appliquent certaines techniques aux étapes avancées du processus de l'appariement de schémas. Par exemple, lors de l'étape de la sélection de l'alignement, une approche *Top-K* (Gal, 2006a) a été adoptée. Cette approche, au lieu de proposer une seule correspondance qui peut être considérée comme la seule meilleure correspondance (*Top-1*), propose un palmarès des meilleures correspondances. La justification de cette approche vient du fait que l'incertitude est inévitable dans un processus d'appariement, car il serait irréaliste de s'attendre qu'un seul apparieur puisse identifier un alignement correct pour tous les scénarios d'appariement. Effectivement, la possibilité qu'un apparieur soit induit en erreur à cause de la confusion sémantique reste une possibilité très élevée.

L'approche basée sur le vote (Villanyi *et al.*, 2014) est une autre contribution pour la réduction de l'incertitude, en passant par l'automatisation de la validation des correspondances suggérées par chaque apparieur linguistique. Cette approche exploite des apparieurs linguistiques, basés sur l'appariement linguistique flou[27], auxquels on assigne des fonctions d'abonnement flou[28]. Le résultat de chaque apparieur est traité comme un vote (similarité proposée) en faveur d'une correspondance donnée. La

[27] De l'anglais « Fuzzy Linguistic Matching »

[28] De l'anglais «*Fuzzy membership function* »

décision finale, qui va être réduite à un problème de combinaison des votes, est basée sur une règle floue pour le calcul de la combinaison des votes.

De l'autre côté, les approches extrinsèques offrent l'avantage d'être utilisées de façon indépendante pour la validation des résultats de divers outils. Elles appliquent diverses techniques, lors de l'étape que l'on peut qualifier comme faisant partie de l'étape de post-traitement, pour la validation de l'alignement. Parmi ces approches, on peut citer une approche basée sur le raisonnement probabiliste (sous incertitude) pour la validation de l'alignement (Castano *et al.*, 2008). Elle se base sur le formalisme *P-SHIQ(D)* pour le raisonnement probabiliste qui permet de détecter la présence d'incohérences dans l'alignement à valider. Aussi, parmi les approches récentes pour la validation de l'alignement, on trouve une solution reposant sur l'externalisation ouverte[29] (Zhang *et al.*, 2013). Cette approche cible la réduction de l'incertitude en d'exploitant une plateforme d'externalisation ouverte (e.g. *Amazon Mechanical Turk*[30]) pour poser des questions sur la validité de l'alignement. L'approche permet une certaine optimisation pour le rapport entre the coût (le nombre de questions posées sur la plateforme) et la performance (la réduction de l'incertitude).

Il est indéniable qu'un consensus se dégage concernant l'aspect inévitable de l'incertitude, quant au résultat de l'appariement, et de la nécessite de composer avec. On voit bien que la multitude des approches, que ce soit intrinsèques ou extrinsèques, pour la gestion de l'incertitude, dénote non seulement de l'importance de cette problématique mais encore laisse paraitre que c'est une problématique qui demeure encore ouverte et qui nécessite encore de la recherche qui permettrait de la réduire avec un effort limité de la part de l'utilisateur. Au-delà des approches discutées plus-haut, visant la réduction de l'incertitude, l'optimisation du processus d'appariement en

[29] De l'anglais « Crowdsourcing »

[30] *Amazon Mechanical Turk (AMT)*, utilisée largement dans le domaine de l'externalisation ouverte

général, et l'optimisation automatique de ce processus en particulier, est une autre avenue pour faire face à ce défi de l'incertitude.

1.2.4.2 Optimisation et complexité

Construire et optimiser un système d'appariement de schémas, est une tâche complexe, manuelle et fastidieuse. Même si souvent, dans des conférences, on annonce des performances satisfaisantes concernant la qualité des correspondances trouvées automatiquement, il n'en demeure pas moins que ces résultats sont souvent le fruit d'un processus d'appariement utilisant une configuration avec des paramètres très finement calibrés. Le calibrage et l'optimisation de ces paramètres requiert souvent une très grande expertise. Il est à noter que certain systèmes fournissent une interface utilisateur qui permet de faciliter la tâche à l'utilisateur pour la configuration et l'optimisation des paramètres du processus d'appariement (Peukert *et al.*, 2012).

Le problème concernant l'optimisation de l'appariement de schémas, peut être posé selon (Lee *et al.*, 2007), à travers la question suivante : « Ayant un scénario d'appariement donné, comment sélectionner les bons composants d'appariement (c.à.d. appariеurs, mesures de similarité, etc.) à exécuter, et comment ajuster la multitude des 'boutons de contrôle', en d'autres termes paramètres (e.g. seuils, coefficient, poids), de ces composants ? »[31]. La réponse à cette question, selon les auteurs, est l'optimisation du processus d'appariement, sans laquelle les systèmes d'appariement échouent souvent à exploiter les caractéristiques du domaine et ainsi produisent des résultats imprécis. Bien que importante, l'optimisation est très difficile, en raison de la multitude de paramètres impliqués, la grande variété des techniques

[31] Traduction libre de l'anglais: "*given a particular matching situation, how to select the right matching components to execute, and how to adjust the multiple "knobs" (e.g., threshold, coefficients, weights, etc.) of the components?*"

utilisées (e.g. base de données, apprentissage machine, théorie de l'information), ainsi que l'interaction complexe entre ces composants (Lee *et al.*, 2007).

La performance du processus d'appariement est tributaire, à chacune de ses étapes, du calibrage d'ensemble de paramètres et de configurations. En effet le choix de différentes stratégies, incluant le choix des mesures de similarité, de seuils, de poids, et de fonctions d'agrégation, influence grandement la qualité du résultat final. La multitude de ces paramètres et de ces combinaisons, rend une recherche exhaustive d'une combinaison optimale, dans l'espace de recherche de toutes les combinaisons possibles, quasi-impossible (i.e. *complexité* ou *explosion combinatoire*) (Rahm et Bernstein, 2001b ; Berkovsky *et al.*, 2005 ; Lee *et al.*, 2007 ; Martinez-Gil *et al.*, 2008).

Sans nul doute que, l'optimisation des différents paramètres du processus d'appariement est une tâche qui requiert un effort non négligeable. Les paramètres les plus fréquemment touchés par l'optimisation sont les poids (e.g. d'une fonction d'agrégation), et les seuils de similarité (e.g. pour la sélection de l'alignement). S'ajoute à cela, l'optimisation des stratégies de sélection et d'exécution des mesures de similarité. L'automatisation de la tâche de l'optimisation vise généralement la réduction de l'effort requis pour celle-ci. Les solutions proposées pour le problème de l'optimisation du processus d'appariement peuvent donc, être classées schématiquement en deux grande catégories : (i) approches manuelles et (ii) approches automatiques.

Tout d'abord, considérons le cas des outils de la catégorie de ceux proposant une optimisation manuelle. Dans cette catégorie, on trouve un des outils les plus populaires dans la littérature, en l'occurrence *COMA* (Do et Rahm, 2002). C'est un outil permettant une configuration manuelle du processus d'appariement. Ce système d'appariement est basé sur des stratégies d'appariement configurables permettant, par exemple, la configuration des stratégies de sélection des apparieurs (parmi une multitude d'apparieurs) ainsi que la combinaison de leurs résultats. (Bernstein *et al.*,

2004) affirment que *COMA* est le premier travail à avoir régler les problèmes d'ingénierie des systèmes d'appariement de schémas[32].

Ensuite, on trouve les outils pouvant être classés comme basés sur une approche d'optimisation automatique. Vu l'importance que revêt l'optimisation dans le contexte de l'appariement de schémas, on trouve dans cette catégorie une pléiade d'outils avec des approches très diverses. Pour commencer, on peut citer, à titre d'exemple, la proposition de (Peukert *et al.*, 2012) qui consiste en une approche avec une capacité d'auto-configuration pouvant adapter le processus dynamiquement aux différents scénarios d'appariement. L'approche proposée se veut être un système d'appariement adaptatif, permettant de produire une bonne qualité d'appariement pour des scénarios d'appariement différents. Cette approche, à partir des schémas en entrée, calcule un ensemble de caractéristiques (*matrice de caractéristiques*). Ces caractéristiques sont ensuite évaluées par des règles d'appariement, pour décider de la sélection des différentes composantes du système d'appariement (e.g. les apparieurs, les fonctions d'agrégation) et ainsi permettre de générer un processus d'appariement complexe pouvant s'adapter, à la volée, aux différents scénarios d'appariement. Les règles qui conditionnent cette faculté d'adaptation du processus d'appariement sont évaluées selon des « *fonctions de pertinence* » qui sont créés maintenues manuellement par l'utilisateur.

Un autre exemple de cette catégorie, serait l'outil *RiMOM* (Li *et al.*, 2009), qui propose une sorte d'auto-optimisation permettant une sélection dynamique de l'apparieur à utiliser. Ce prototype d'appariement d'ontologies est l'un des premiers systèmes à implémenter une sélection dynamique des apparieurs. Les éléments de schémas et leurs instances sont d'abord appariés sur une base linguistique, et ensuite l'appariement structurel est appliqué seulement si des similarités structurelles sont suffisantes.

---

[32] **De l'angais** (*engineering issues of a schema matching system*)

Plusieurs méthodes linguistiques existent pour l'appariement incluant l'utilisation de documents virtuels pour un élément qui consiste en son nom, ses commentaires et les instances de l'élément.

Toujours à propos des outils disposant d'une capacité d'optimisation automatique, on trouve *YAM* (*Yet Another Matcher*) et *YAM++* (Duchateau *et al.*, 2009 ; Ngo et Bellahsene, 2012). C'est une approche permettant de générer des appariements « à la carte » pour un scénario d'appariement donnée. L'approche *YAM* est basée sur l'apprentissage machine pour l'optimisation des paramètres du processus d'appariement (i.e. entre autres choix des mesures de similarité) en fonction des préférences de l'utilisateur notamment des paramètres concernant la validation des prédictions (e.g. la précision et le rappel), ainsi que les données d'entrainement (un ensemble de correspondances données par l'expert pour un domaine d'intérêt). Les auteurs définissent leur approche non pas comme un autre outil d'appariement de schémas (découverte de correspondances), mais plutôt comme un générateur d'appariements (fabrique d'appariements). *YAM++* se veut être une extension de *YAM* pour le traitement des ontologies.

eTuner (Sayyadian *et al.*, 2005 ; Lee *et al.*, 2007) est un autre outil populaire pour l'optimisation automatique du processus d'appariement en l'occurrence. Cette approche est basée sur la recherche de la meilleure configuration de paramètres (e.g. poids, seuil) pour un ensemble de scénarios synthétiques dont le résultat est connu d'avance (correspondances données par l'expert). Elle nécessite, dans un premier temps, la génération d'un ensemble de schémas synthétiques, à partir d'un schéma initial, sémantiquement équivalents, et ce en injectant des perturbations selon des règles prédéfinies prise d'une manière aléatoire (par exemple, remplacer le mot « *EMPLOYEES* » par « *EMP* »). Dans un deuxième temps, l'outil d'appariement, dont on désire optimiser les paramètres, est exécuté pour apparier le schéma initial et l'ensemble des schémas synthétiques et ce, en ajustant les paramètres à chaque

itération, et en comparant le résultat obtenu avec le résultat attendu jusqu'à trouver la configuration optimale (les valeurs des paramètres permettant à l'outil de trouver des correspondances similaires à celles attendues par l'utilisateur). Comme l'espace des différentes valeurs de configuration, permettant de trouver la configuration optimale, est très grand, rendant une recherche exhaustive impraticable, *eTuner* adopte une approche gloutonne appelée « optimisation par étape[33] ». D'abord, il faut rappeler que *eTuner* n'est pas un outil d'appariement mais un outil permettant l'optimisation des outils d'appariement. Deuxièmement, cet outil nécessite un travail préalable de la part de l'humain qui n'est pas négligeable, comme par exemple, préparer les règles qui introduisent les perturbations, valider que les schémas synthétiques sont sémantiquement équivalents au schéma initial ou encore fournir au préalable les correspondances escomptées. Enfin, la performance escomptée de l'outil d'appariement, après le travail d'optimisation de *eTuner*, reste largement tributaire du choix de l'outil d'appariement à optimiser.

Et enfin, comme dernier exemple, on peut citer une autre approche (Berkovsky *et al.*, 2005), basée sur les algorithmes génétiques pour l'optimisation de la combinaison des résultats des différents apparieurs. Cette approche considère l'appariement de schémas comme une problématique de recherche que l'on peut résoudre par l'application d'un algorithme génétique. En effet, l'algorithme a pour but de rechercher la meilleure solution en termes de combinaison du résultat de différents apparieurs en utilisant différents poids. Le succès d'un algorithme génétique, selon les auteurs, dépend de la « *fonction d'évaluation*[34] » qui dirige la recherche vers des chemins prometteurs. La fonction d'évaluation pour cette approche utilise la mesure de la précision (proportion de solutions trouvées qui sont pertinentes).

---

[33] De l'anglais « *staged tuning* »

[34] De l'anglais « *fitness function* »

Malgré tous les efforts et les avancées pour augmenter la qualité de leurs résultats, les approches actuelles essuient quand même plusieurs critiques dans la littérature. Parmi les critiques les plus récurrentes, on trouve la complexité des solutions, l'inefficacité de l'optimisation de ces solution face aux changements des scénarios d'appariement et enfin l'effort élevé nécessaire pour arriver à des résultats satisfaisants. Par exemple, (Peukert *et al.*, 2012) affirment que les solutions actuelles ne sont pas assez robustes pour être en mesure de faire face aux différents scénarios d'appariement, sans un effort élevé de configuration. La tâche de configuration et d'optimisation d'une solution existante d'appariement de schémas peut encore sembler, dans certains cas, plus complexe que la reconstruction de la solution à partir de zéro. Face à la complexité cumulée à différents niveaux, selon (Yatskevich, 2003 ; Shvaiko *et al.*, 2008), les approches proposées ne mettent de l'avant, d'une manière générale, que des solution dédiées à résoudre des défis spécifiques. Quant à (Peukert, 2013), il annonce qu'à cause de leurs limites, les solutions proposées sont jusqu'au jour d'aujourd'hui, peu utilisées dans l'industrie. Et enfin, pour (Gal et Shvaiko, 2009), l'auto-optimisation pour les systèmes d'appariement de schémas reste un champs largement inexploré.

1.2.5 Synthèse

En guise de synthèse, on peut dire que le processus d'appariement peut être réduit à trois étapes importantes : (i) l'étape de la sélection et de l'exécution des apparieurs, (ii) l'étape de la combinaison des résultats, et enfin (iii), l'étape de la sélection de l'alignement.

Comme le montre la figure 1.7 les facteurs impactant l'appariement de schémas sont :

- Hétérogénéité : En général, comme la tâche de l'appariement implique la sémantique (compréhension du contexte) pour avoir une certitude complète sur la qualité du résultat, l'implication humaine est requise. Le défi principal dans tous les cas d'appariement automatique consiste à décider de la bonne correspondance. C'est une tâche très difficile à cause principalement de l'hétérogénéité des données.

- Incertitude : l'hétérogénéité cause de l'incertitude sur le résultat final de l'appariement. La raison de cette incertitude réside principalement dans l'ambiguïté et l'hétérogénéité, tant au niveau syntaxique que sémantique, qui souvent caractérisent les schémas à apparier.

- Optimisation : l'incertitude sur la qualité du résultat implique l'optimisation de du processus pour améliorer la qualité. Tester différentes combinaisons (e.g. différentes mesures, fonctions d'agrégation, sélection des correspondances). Chaque étape du processus d'appariement implique le choix entre plusieurs stratégies ce qui conduit à une explosion combinatoire (complexité).

- Complexité : l'optimisation génère de la complexité à cause de l'espace de recherche (explosion combinatoire). De plus, le changement des scénarios d'appariement vient aggraver cette complexité dans la mesure ou le résultat de l'optimisation devient souvent obsolète avec le changement de scénarios.

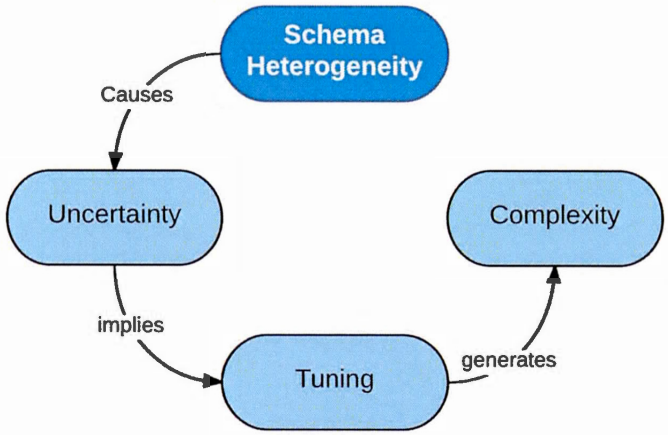


Figure 1.7 Causes et effets relatifs à l'appariement

Relativement aux changements des scénarios d'appariement, nous avons montré plus haut, que les approches existantes, que ce soit les approches pour le processus d'appariement en tant que tel ou bien les approches pour son optimisation, et malgré la multiplication des techniques, montrent des limites quand il s'agit de s'adapter à tous les contextes d'appariement. Par conséquent, elles peinent à produire des résultats d'une bonne qualité pour des différents scénarios d'appariement.

Un des points en commun entre toutes les approches existantes est le mode de pensée derrière ces approches à savoir la pensée *réductionniste* (par opposition au *holisme*). Cette pensée *réductionniste* engendre pour les approches existantes des caractéristiques fondamentales et intrinsèques conduisant aux limitations dont elles pâtissent aujourd'hui.

1.3 Réductionnisme des approches existantes

Dans cette section nous allons nous attarder sur les principales corrélations que l'on peut établir, de notre point de vue, entre certaines caractéristiques intrinsèques aux approches actuelles et les limitations auxquelles ces approches font face (e.g. réduction de l'incertitude).

La pensée *réductionniste* est à l'origine des caractéristiques qui sont, à notre sens, à la racine des causes qui empêchent l'appariement automatique de schémas de faire face complètement aux défis et difficultés (présentés dans la section précédente).

Le *réductionnisme* - par opposition à la *systémique* ou au *holisme*- est un concept philosophique qui renvoie aussi bien au mode de pensée de ces solutions qu'à leur méthodologie de modélisation (*analytique*). Le réductionnisme prône la réduction de la complexité des systèmes ou des phénomène à leur éléments fondamentaux qui seraient alors plus simples à comprendre et à étudier (Fortin, 2005). En biologie par exemple, pour comprendre le fonctionnement d'un système biologique complexe (e.g. corps humain) on a besoin de le réduire à la compréhension de ces constituants (e.g. les organes internes, le système nerveux, le cerveau, etc.) et à la compréhension de simples relations de causes à effets. Cette approche réductionniste, malgré sa grande efficacité dans plusieurs domaines, montre toutefois des limites pour certains sujets. En effet, une abstraction de la réalité à une linéarisation de simples relations de causes et d'effets entre des constituants fondamentaux d'un système complexe apparait, pour l'explication de certains phonèmes ou la résolution de certains problèmes, hautement simplificatrice. Pour mettre en exergue cette simplification limitatrice, prenons par exemple le cerveau humain et posons-nous la question suivante : est-ce qu'en ramenant le cerveau humain a un ensemble de neurones échangeant des signaux électriques on pourrait être capable d'expliquer un phénomène émergent tel que la conscience ?

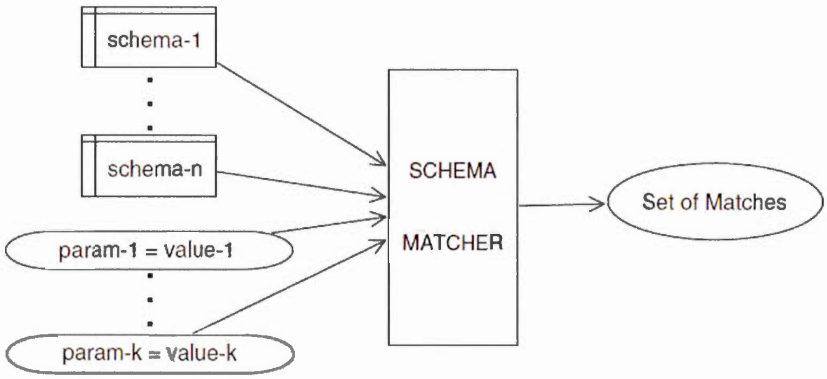


Figure 1.8 Processus d'appariement **(Bellahsene et Duchateau, 2011)**

Concernant l'appariement de schémas, il apparait clairement, comme peut le montrer la figure 1.8, que toutes les approches actuelles suivent la pensée réductionniste. Elles ramènent le processus de schéma, à une fonction linéaire avec un ensemble d'intrants et d'extrants. Cette fonction peut se décomposer en une série de modules, dont chacun est responsable du déroulement d'une étape du processus (e.g. sélection et exécution des appariеurs) et se constituant d'éléments plus au moins simples (e.g. appariеurs, fonctions, paramètres). Les tentatives de résolution des difficultés rencontrées par l'appariement automatique de schémas (e.g. incertitude), suivent le même schéma de pensée à savoir la focalisation sur un aspect bien précis du fonctionnement du processus et ensuite la tentative d'optimisation de ce module ou de la composante responsable de cet aspect. À titre d'exemple, pour augmenter la qualité du résultat, l'approche *YAM* vise seulement l'optimisation de l'aspect concernant la sélection des appariеurs et propose pour ce faire une approche basée sur les techniques d'apprentissage machine permettant de générer une sélection d'appariеurs adaptés pour un scénario d'appariement donné.

À l'opposé de l'approche réductionniste, on trouve l'approche holistique et systémique qui va s'intéresser au tout et à ses qualités émergentes. L'hypothèse, que nous défendons dans cette thèse, est que la problématique de l'appariement automatique de

schémas mériterait qu'elle soit abordée dans son entièreté et vue comme un système complexe comprenant plusieurs composantes dont la simulation des interactions non-linaires de ces composantes permettrait l'émergence du résultat final.

Les caractéristiques profondes et communes, à tous les systèmes actuels d'appariement, pouvant partiellement expliquer leur incapacité à surmonter la limitation de la complexité de l'appariement et ainsi de répondre à d'autres défis tel que l'incertitude, peuvent être déclinées comme suit : ces systèmes sont (i) *compliqués* et non pas *complexes*, , (ii) *linéaires* (analytiques, déterministes, prévisibles) et non pas *non-linéaires*, (iii) *centralisés* et non pas *décentralisés* (parallélisme, résolution par émergence), (iv) et enfin *configurables* et non pas *adaptables* (auto-configuration, auto-optimisation).

1.3.1 Complication

Commençant tout d'abord par la première caractéristique des solutions actuelles à savoir la *complication* et non la *complexité*.

Avant de développer ce point, il serait pertinent de citer une opposition intéressante, avancée par (Le Moigne, 1990), entre un système compliqué et un système complexe quant à leur intelligibilité : « Pour comprendre et donc pour donner du sens à un système compliqué, on peut le simplifier pour découvrir son intelligibilité (explication). Un système complexe, on doit le modéliser pour construire son intelligibilité. ».

Un système complexe, à partir de règles simples peut produire des résultats étonnamment complexes, contrairement aux systèmes compliqués ou la résultante des algorithmes est proportionnelle à la nature de l'algorithme (e.g. algorithme simple donne des réponses simples et prédictibles). Cela étant dit, dans le but d'améliorer la qualité du résultat (réduire l'incertitude), les approches actuelles vont appliquer des techniques et des stratégies, en aval, en passant généralement par la sélection ou la

validation de l'alignement, et/ou en amont, en passant par l'optimisation du processus d'appariement (stratégies, paramètres, etc.). Dans les deux cas, les systèmes d'appariement automatique accumulent une grande complication (deviennent compliqués). À l'inverse, les systèmes complexes promettent la résolution de problèmes complexe, tel que l'appariement automatique de schémas, avec des règles simples. Ce qui nous renvois a un postulat (Le Moigne, 1990) qui, d'une certaine manière, résume bien cette situation : « La simplification du compliqué appliqué au complexe a pour conséquence une aggravation de la complexité par mutilation et non pas la résolution du problème considéré. »

L'optimisation des différents paramètres et facteurs influençant le résultat d'un processus automatique d'appariement est une tâche complexe : dans le sens que cela requiert de la part de l'utilisateur une expertise du système d'appariement (le processus avec ses différentes composantes) et un effort accru.

Comme on l'on déjà mentionné, on distingue deux types de solutions d'optimisation. Le premier type concerne les solutions intrinsèques cherchant à s'appuyer sur l'adaptation (auto-optimisation, auto-configuration) et le second type concerne les solutions extrinsèques s'appuyant sur les outils d'optimisation tel que *eTuner*.

Pour les solutions intrinsèques, la principale critique serait que même si le but avoué est de réduire l'incertitude avec un effort minimum, reste que l'expertise requise pour la maintenance de ces outils est très élevée (dans la phase de conception ou de maintenance). Cet effort peut devenir très rapidement un frein à l'utilisation du système. Prenons à titre d'exemple le système *COMA* qui permet de bâtir de nouvelles stratégies d'optimisation (en plus de celles proposées par défaut) par l'utilisateur (i.e. un genre de script avec une syntaxe propre à *COMA*). Cet exercice de création de nouvelles stratégies (i.e. la sélection des composants, le calibrage, les tests) même s'il peut s'avérer bénéfique pour un certain nombre de scénarios, peut-être à refaire pour un autre type de scénarios (e.g. changement de domaine d'affaire). Concernant

l'approche (Peukert *et al.*, 2012), on a vu plus haut, qu'elle avait besoin que l'utilisateur ait la délicate tâche de concevoir et de maintenir des « *fonctions de pertinence* ». Prenons un autre exemple *YAM* (Duchateau *et al.*, 2009), l'exercice qui consiste à faire de l'apprentissage machine (s'appuyant sur des exemples étiquetés pour créer un modèle de classification) doit être refait pour chaque nouveau domaine. Prenons un autre exemple, *eTuner*, la création de scénarios synthétiques pour chaque nouveau scénario d'appariement est aussi la responsabilité de l'utilisateur. On voit bien que l'utilisateur doit toujours être implique et un haut niveau d'expertise est souvent requit de la part de ce dernier.

1.3.2 Linéarité

La seconde caractéristique qui marque les solutions existantes c'est la *linéarité*. Cette linéarité touche à la fois la causalité entre causes et effets mais aussi la proportionnalité entre les entrants et les extrants. « *La causalité linéaire est conçue comme extérieure aux objets : c'est une causalité supérieure où les mêmes causes, dans les mêmes conditions, entraînent toujours les mêmes effets. Cause et effet existent dans un rapport de subordination : l'effet, tout-dépendant, obéit mécaniquement à la cause, toute-puissante.* » (Fortin, 2005, p. 46).

La linéarité des systèmes d'appariement existants, est à la source du *déterminisme* et de la prévisibilité qui caractérise leurs résultats. Cela étant dit, on est en droit de se demander, et à juste titre d'ailleurs, en quoi le déterminisme peut avoir un impact sur les défis de l'appariement, notamment l'incertitude.

Signalons à ce propos une observation, qui nous a interpellé lors de l'évaluation des outils existants concernant la réplication exacte du résultat de l'appariement, y compris quand ce résultat comporte des correspondances erronées, et cela malgré la multiplication des exécutions. En d'autres termes, avec les mêmes paramètres en entrée (i.e. schémas en entrée, préférences utilisateurs, configuration), on obtient toujours le même résultat en sortie. Cette observation peut sembler triviale, pourtant, elle met en

relief une caractéristique fondamentale, à savoir la *linéarité* qui conduit au *déterminisme*, et la *prévisibilité* (avec un comportement monotone et prévisible), et qui constitue une des différences majeures entre l'approche qui fait l'objet de cette thèse et les approches existantes. En effet, les modèles derrière les approches actuelles d'appariement sont essentiellement des modèles déterministes. Ces modèles commencent par effectuer des calculs de similarité entre les éléments des deux schémas avec des mesures de similarité dont le choix est fait à priori, puis créent une matrice de similarités et enfin produisent le résultat. Même les approches dont la sélection des mesures de similarité est dynamique, tel que l'outil *YAM* (Duchateau *et al.*, 2009), restent au final des modèles centralisés et déterministes car une fois l'optimisation des appariеurs faite (stratégie combinant des mesures), pour un scénario d'appariement, ces mesures choisies comme optimales ne changent plus.

En partant de cette observation, nous avons été fortement motivés à explorer une nouvelle avenue dans le domaine de l'appariement automatique de schémas. Cette nouvelle avenue pourrait être résumée par la question suivante :

Est-il possible de résoudre le problème de l'appariement automatique de schémas par une approche non- linéaire basée sur un modèle stochastique (aléatoire), non prévisible et sans contrôle centrale ou chaque expérimentation peut donner lieu, même avec les mêmes données et paramètres en entrée, à des résultats potentiellement différents ?

On pense qu'une grande partie de la réponse peut se baser sur une intuition, à la fois simple et fondamentale, qui consiste à imaginer les éléments des schémas dont on désire trouver les correspondances comme des entités autonomes capables de trouver les meilleurs appariements (correspondances), entre elles et celles du groupe opposé (schémas opposé). Contrairement aux approches existantes qui considèrent les schémas en entrée comme un ensemble d'éléments statiques à partir desquels, un processus linéaire et déterministe va générer un autre ensemble de correspondances.

Ce déterminisme touche même les approches qui se réclament comme adaptatives. L'approche (Peukert *et al.*, 2012), par exemple, qui s'adapte dynamiquement au scénario d'appariement, en se basant sur des règles, demeure un système déterministe produisant les mêmes résultats malgré la multiplications des exécutions.

1.3.3 Centralisation

Les approches actuelles de l'appariement de schémas suivent un modèle de traitement centralisé - par opposition à un modèle décentralisé ou réparti- pour le contrôle et la prise de décision concernant la résolution du problème de l'appariement. Le modèle centralisé est un modèle monolithique et hiérarchique qui impose la résolution du problème d'une manière directive monodirectionnel (du bas vers le haut).

À l'opposé du modèle centralisé, le modèle décentralisé tel que « *intelligence en essaim*[35] », auquel on accole aussi de nombreuses autres dénominations, notamment « *intelligence collective* », « *intelligence artificielle distribuée* », « *résolution distribuée de problèmes* », etc., est basé sur le principe de la collégialité de la résolution des problèmes (faire émerger une solution globalement optimale à partir de solutions localement optimales).

La notion de « *intelligence en essaim* » a été définie par l'un des premiers à avoir introduit ce terme (Beni, 2009), comme suit : « *The intuitive notion of "Swarm Intelligence" is that of a "swarm" of agents (biological or artificial) which, without central control, collectively (and only collectively) carry out (unknowingly, and in a somewhat-random way) tasks normally requiring some form of "intelligence"* ». Il est important de souligner qu'outre le concept de l'intelligence décentralisée, cette définition fait ressortir un autre concept fondamental à la notion d'intelligence collective, à savoir la notion de *l'aléatoire* (*stochasticité*).

---

[35] De l'anglais « *swarm intelligence* »

Pour comprendre, dans le contexte de l'appariement de schémas, en quoi la centralisation du traitement peut être un désavantage, commençons tout d'abord par examiner le lien entre la centralisation du traitement et un aspect caractérisant toutes les solutions actuelles d'appariement, à savoir la *complication* (à ne pas confondre avec la *complexité*).

On pense que le haut degré de complication dont souffre les solutions d'appariement actuelles, les rendant par le fait même très exigeantes en termes d'effort et d'expertise humaine, a un lien direct avec le mode de contrôle centralisé et hiérarchique du processus d'appariement. Dans le but d'aboutir à un processus d'appariement performant - avec des stratégies optimales- pouvant donner lieu à un résultat de qualité avec un minimum d'incertitude, ce mode d'organisation du traitement conduit, et ce dès la conception du système d'appariement, à un empilement de techniques et de stratégies rendant souvent l'algorithme d'appariement très compliqué à concevoir et à maintenir sans pour autant répondre complètement aux défis de l'appariement. Prenons par exemple le défi de l'optimisation de l'appariement, comme on l'a déjà vu, les solutions proposées sont dans l'incapacité, à cause de l'explosion combinatoire, d'explorer tout l'espace de recherche des combinaisons possibles, avec l'objectif de sélectionner la meilleure combinaison - d'apparieurs, des stratégies et des paramètres pour un scénario d'appariement donné – pouvant permettre une amélioration du résultat de l'appariement.

Par ailleurs, l'aspect central du traitement du processus d'appariement, ne favorise pas le parallélisme du traitement.

En revanche, Les systèmes basés sur la répartition de l'intelligence, disposent de la capacité de faire émerger, sans contrôle externe ou mécanisme de coordination central et de manière totalement distribuée, des solutions novatrices, surprenantes et totalement imprévisibles. Ils permettent une répartition du contrôle et de la prise de

décision sur de nombreux éléments s'exécutant en parallèle sur une ou plusieurs plateformes d'exécution distribuées.

1.3.4 Non-adaptation

En ce qui a trait aux limites de la configuration et de l'optimisation des approches actuelles, on peut principalement relever l'incapacité de ces approches à totalement s'adapter aux perturbations qui peuvent êtres causées par la nouveauté que peut apporter de nouveaux scénarios d'appariement. L'optimisation donne souvent des approches qui ne peuvent pas nécessairement être appliquées à des scénarios d'appariement complètement nouveaux (pour lesquels les paramètres issus de l'optimisation deviennent inefficaces ou inadaptés). De plus, ces approches proposent des configurations prédéterminées par défaut (stratégies de sélection d'apparieurs ou stratégies de combinaison des résultats) qui ne conviennent pas à tous les scénarios et par conséquent n'arrivent toujours pas à garder une certaine robustesse devant l'imprévisibilité que peut apporter de nouveaux scénarios d'appariement. Cela nécessite une capacité d'adaptation (auto-configuration, auto-optimisation) pour faire face à cette nouveauté, sans pour autant recourir à l'optimisation manuelle.

Pour les rares outils qui disposent de capacités d'adaptation (auto-configuration), ils ne jouissent que d'une capacité d'adaptation limitée touchant uniquement certaines parties du processus (sélection et combinaison des apparieurs, e.g. *YAM*) et dont la logique est fixée dans le code. Effectivement, quelques tentatives pour rendre certaines parties des systèmes d'appariement plus adaptatives et plus auto-optimisables (Hu et Qu, 2008 ; Li *et al.*, 2009 ; Mao *et al.*, 2008 ; Pirró et Talia, 2010) ont été proposées dans la littérature. Par exemple, le *RiMOM*, système d'alignement d'ontologies (Li *et al.*, 2009), adapte dynamiquement le choix de l'apparieur entre un apparieur terminologique (basé sur le nom de l'élément) ou un apparieur structurel (basé sur la structure du schéma) et ce en calculant des propriétés à partir des schémas d'entrée. Cependant, ces comportements d'adaptation sont fixés dans le code et ne permettent de faire face qu'à

une petite partie du problème d'optimisation du processus d'appariement (Peukert *et al.*, 2012).

L'approche proposée par (Peukert *et al.*, 2012) se veut être un système d'appariement adaptatif avec une capacité d'auto-configuration permettant au processus de s'ajuster dynamiquement à différents scénarios d'appariement. Le mécanisme d'adaptation de cette approche, comme on l'a déjà mentionné, se base sur des règles basées sur des « *fonctions de pertinence* » qui sont créés est maintenues manuellement par l'utilisateur. On peut encore une fois parler d'adaptation limitée dans la mesure où ces règles et ces fonctions sont définies manuellement, ce qui *in fine* doit limiter leur évolution (maintenabilité) et ainsi limiter l'adaptation. En effet, le cœur du module permettant l'adaptation est basé sur des fonctions qui certes non sont pas fixée dans le code mais fixée manuellement lors de la conception (pas d'apprentissage automatique). Or pour parler d'adaptation, il faut justement s'éloigner tant que possible du manuel et aller vers des solutions favorisant l'adaptation par l'évolution et l'apprentissage automatique (auto-configuration et auto-optimisation)

1.3.5 Discussion

La pensée *réductionniste* derrière toutes les approchent actuelles leur confère des caractéristiques intrinsèques conduisant aux limitations dont elles pâtissent. Il convient de rappeler ces caractéristiques intrinsèques : caractéristiques ces systèmes sont (i) compliquées (et non pas complexes), (ii) réductionnistes (et non pas holistiques), (iii) linéaires (et non pas non-linéaires), (iv) centralisées (et non pas décentralisées), (v) et enfin configurables (et non pas adaptables).

Ces caractéristiques intrinsèques, sont à notre avis, la cause profonde qui fait que les approches actuelles éprouvent, d'un côté un manque d'autonomie (dans la mesure où l'implication de l'expert est requise, d'une manière ou d'une autre, pour la configuration et l'optimisation du processus d'appariement) et d'un autre coté un manque d'adaptation (dans le sens où cet exercice d'optimisation doit être refait à

chaque fois pour l'adapter aux différents types de scénarios rencontrés par le processus d'appariement).

La nécessite d'explorer de nouvelles approches dans le but d'apporter des réponses systémiques et holistiques à la problématique de l'appariement nous amène à soulever la question suivante : comment peut-on avoir des solutions d'appariement résilientes aux perturbations indues par les changements des scénarios d'appariement de schémas ?

Notre postulat est qu'une bonne partie de la réponse peut venir de la théorie des systèmes complexes adaptatifs où la modélisation de la complexité, de l'adaptation et de l'évolution des systèmes est au cœur même de cette théorie. Avoir un processus d'appariement de schémas qui peut faire face aux défis de l'appariement automatique de schémas nécessite, à notre sens, un changement de paradigme, en plaçant l'adaptation, l'évolution et l'auto-organisation comme principales préoccupations. Nous croyons que les systèmes adaptatifs complexes, qui ont été utilisés pour expliquer certains phénomènes biologiques, sociaux et économiques, peuvent être la base d'un paradigme de programmation pour les outils d'appariement de schémas.

Il nous apparaît assez clairement que les systèmes complexes adaptatifs pourraient nous apporter l'adaptation (auto-configuration et auto-optimisation) qui devrait décharger l'utilisateur de la complexité et de l'effort lié à la configuration et à l'optimisation des systèmes d'appariement automatiques de schémas.

Dans le contexte de l'appariement automatique de schémas, on entend par l'adaptation, la capacité du système à faire : (i) l'auto-configuration[36] et (ii) l'auto-optimisation[37].

- L'auto-configuration : c'est la capacité de se configurer de façon dynamique. Un système d'appariement peut s'adapter immédiatement, sans intervention manuelle, au déploiement de nouveaux composants (e.g. nouveaux apparieurs) ou aux changements des scénarios d'appariement.
- L'auto-optimisation : elle désigne la capacité du système d'appariement à maximiser efficacement les paramètres et stratégies pour le processus d'appariement avec, encore une fois, aucune intervention manuelle. À cet égard, l'auto-optimisation traite principalement du problème de la complexité engendrée par l'explosion combinatoire.

Dans la dernière partie de ce chapitre, nous tenterons de jeter la lumière sur les systèmes complexes adaptatifs, et ce en faisant un survol de ce paradigme ainsi que d'une des méthodes de modélisation permettant de modéliser un système complexe adaptatif, en l'occurrence la modélisation et la simulation à base d'agents.

1.4 Systèmes complexes adaptatifs

1.4.1 Concepts

1.4.1.1 Systémique

Les systèmes complexes adaptatifs trouvent une de leurs multiples racines dans la pensée systémique (théorie des systèmes). La systémique prône une approche holistique, contrairement à une approche réductionniste, pour la compréhension d'un système. En effet, pour la systémique le tout est plus que la somme des parties. Cette

[36] De l'anglais « *self-configuration* »

[37] De l'anglais « *self-optimization* »

notion de totalité implique comme démarche, pour la compréhension du fonctionnement d'un système, l'étude non seulement de l'ensemble des composants du système mais aussi les interactions de ces composants entre eux (Bertalanffy, 1968).

1.4.1.2 Complexité

Selon (Jones, 2003), il y aurait deux grandes distinctions à faire lorsqu'il s'agit de comparer les systèmes : (i) la première distinction serait : « adaptif » vs « déterministe ». Les systèmes adaptatifs sont constitués d'agents qui ont un lien avec leurs voisins et qui répondent aux changements avec un certain degré de liberté (c.à.d. répondre avec des règles simples). Un système avec une telle structure produit des réponses qui ne sont pas déterminées et qui peuvent être hautement non-linéaires. En revanche, ce qui caractérise les systèmes compliqués, c'est que ces systèmes sont déterministes et prévisibles dans la mesure où la relation entre les données en entrée et en sortie est linaire. (ii) La deuxième distinction serait : « *complexe* » vs. « *compliqué* ». Comme le montre la figure 1.9, dans les systèmes complexes, les relations entre les agents (i.e éléments du système) sont plus importantes que les agents eux-mêmes, contrairement aux systèmes compliqués, où les éléments et leurs relations ont une importance équivalente. Ensuite, les règles simples de systèmes complexes produisent des réponses complexes et adaptatives, contrairement aux systèmes compliqués où la résultante des algorithmes simples se trouve être des réponses simples et prédictibles. Et enfin, dans les systèmes complexes les agents ont la latitude de répondre selon les limites des règles par opposition aux systèmes compliqués où la réponse des composants est complètement déterminée.

Par exemple, un avion est un système compliqué mais non pas un système complexe. En effet, tout le monde peut s'accorder pour dire qu'un avion doit être un système complétement prévisible. À chaque fois que le pilote tire le manche en arrière, l'avion doit prendre de l'altitude et vice-versa. La fourmilière d'une colonie de fourmis, par contre, représente un système complexe dans lequel les fourmis sont reliées entre elles,

agissent selon des règles simples (avec une certaine latitude) et où l'existence des fourmis individuellement n'a pas d'importance par rapport au tout que représente la colonie (la somme de toutes les relations des fourmis de la colonie).

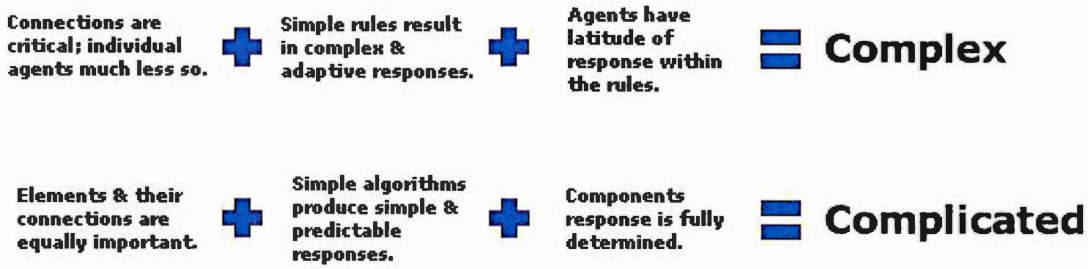


Figure 1.9 « Complexe » vs. « Compliqué » **(Jones, 2003)**

Les systèmes linéaires et déterministes produisent des résultats contrôlables et prévisibles. Les systèmes complexes peuvent produire des résultats novateurs, créatifs et émergents.

Avant de définir ce qu'est un système complexe adaptatif, commençons d'abord par citer quelques caractéristiques intéressantes, avancées par (Simon, 1990), permettant de faire le distinguo entre un système simple et un système complexe : « *(i) Les systèmes qui ont beaucoup de composants peuvent être considérés comme complexes en comparaison des systèmes qui en ont peu. Ainsi la cardinalité d'un ensemble peut être prise comme une mesure de sa complexité. (ii) Les systèmes dans lesquels il y a beaucoup d'interdépendances entre les composants sont généralement considérés comme plus complexes que les systèmes avec moins d'interdépendance entre les composants. (iii) Les systèmes dont le comportement est considéré comme 'indécidables' peuvent être considérés comme complexes comparés à ceux dont le comportement est tenu pour déterminable* ».

Si l'on revient à l'étymologie du mot « complexe », on s'aperçoit qu'elle vient du mot latin « *complexus* » qui désigne ce qui est entrelacé, imbriqué. De ce point de vue, les

systèmes deviennent complexes lorsque la multiplicité des éléments les constituant ainsi que leurs interactions (avec des effets de rétroaction) commencent à exhiber un niveau élevé d'entrelacement et d'imbrication à un point tel que la compréhension du système devient opaque aux méthodes analytiques classiques.

Il serait opportun de signaler, à ce stade, qu'il n'existe pas une définition formelle pour définir un système complexe adaptatif, malgré la multiplication des définitions intéressantes que l'on peut trouver dans la littérature à ce sujet. À ce propos, dans leur livre (Bourgine *et al.*, 2008), abordent la structure des systèmes complexes adaptatifs[38], comme : « *Structurés sur plusieurs niveaux d'organisation, composés d'entités hétérogènes elles-mêmes complexes* ». Sur la dynamique de ces systèmes, les auteurs affirment que : « *Les systèmes complexes, depuis les objets nanoscopiques de la physique et de la biologie jusqu'à l'écosphère, résultent de processus d'émergence et d'évolution : les interactions individuelles engendrent des comportements collectifs qui peuvent manifester des structures organisées. Ces structures émergentes influencent en retour les comportements individuels. Les causes sont multiples et la causalité fonctionne à la fois de façon ascendante et descendante entre les niveaux d'organisation.* ».

La théorie des systèmes complexes adaptatifs trouve ses racines dans diverses disciplines telles que systémique, la cybernétique, etc. La carte des sciences de la complexité (Castellani, 2014a, 2014b) retrace les racines de la théorie de la complexité.

[38] De l'anglais (*Complex Adaptive Systems*)

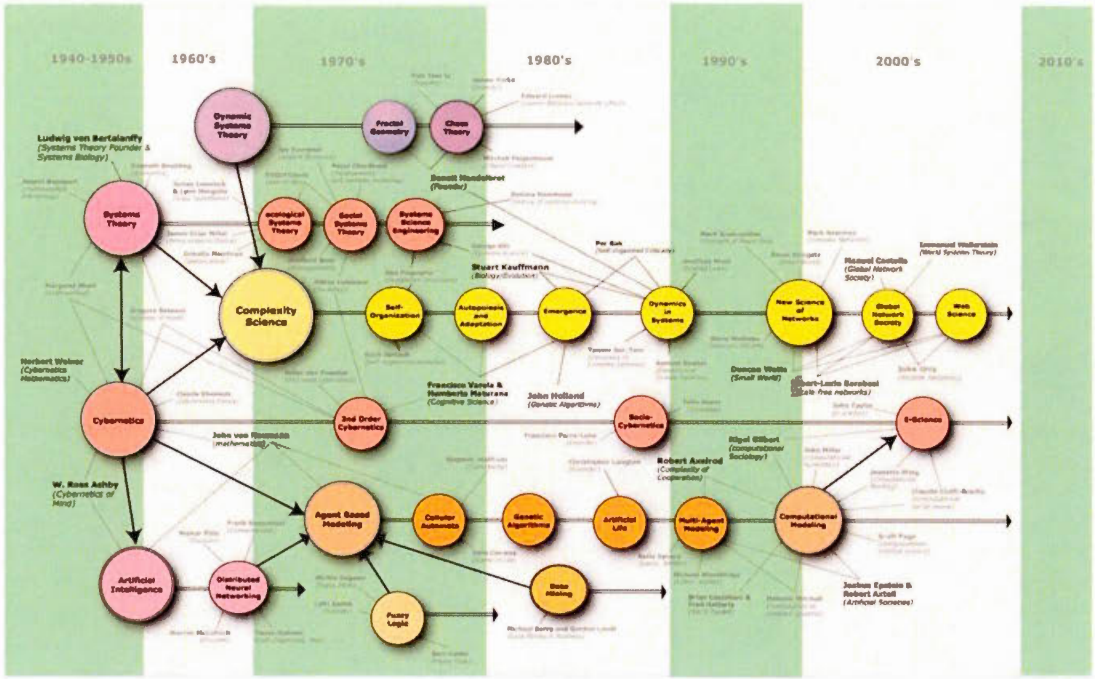


Figure 1.10 Carte de sciences de la complexité **(Castellani, 2014a, 2014b)**

Selon (Holland, 2006), la résolution de beaucoup de problèmes contemporains difficiles, passe par les systèmes complexes adaptatifs. Ces systèmes sont constitués d'un grand nombre de composants, souvent appelés agents, qui interagissent, s'adaptent et apprennent. Une courte liste des problèmes dans lesquels les systèmes complexes adaptatifs soulignent leur omniprésence, évoque la prédiction des changements dans le commerce mondial, l'aide à la compréhension des marchés, l'aide à la préservation des écosystèmes ou encore le contrôle des virus et des spams au niveau d'Internet.

1.4.2 Caractéristiques

Comme il n'existe pas de définition formelle pour les systèmes complexes adaptatifs, dans la littérature, on ne trouve pas une liste unifiée recensant toutes les caractéristiques de ces systèmes. Cela étant dit, il existe cependant, un quasi-consensus sur certaines caractéristiques, que l'on va détailler plus loin, considérées communes aux systèmes

adaptatifs complexes, à savoir : (i) la non-linéarité, (ii) l'adaptation et (iii) l'auto-organisation et l'émergence.

Le diagramme suivant (Andrus, 2005), en plus des caractéristiques listées plus-haut, met en relief la caractéristique de la rétroaction (négative ou positive) qui peut avoir une influence certaine sur le comportement global du système.

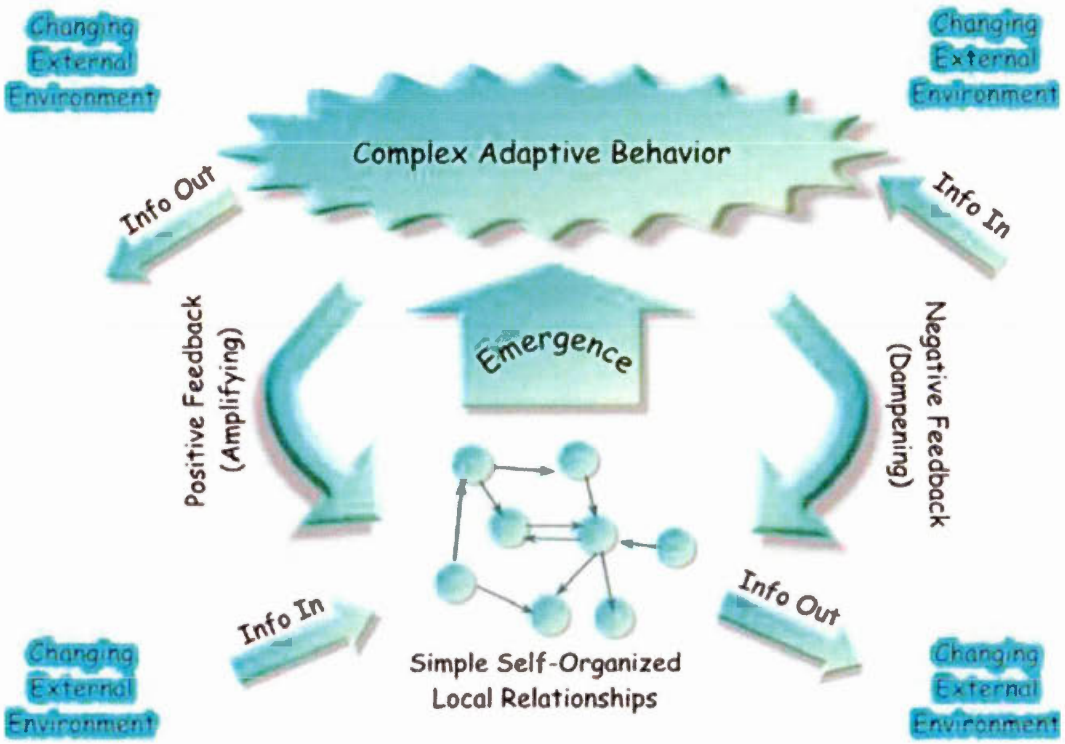


Figure 1.11 Systèmes Complexes Adaptatifs **(Andrus, 2005)**

(Holland, 2006) quant à lui, en plus des caractéristiques communes, met l'accent sur d'autre caractéristiques telles que le parallélisme, l'action conditionnelle ou encore la modularité. Il affirme que malgré des différences substantielles, les systèmes complexes adaptatifs partagent quatre caractéristiques principales, à savoir : (i) *parallélisme*. En effet, les systèmes complexes adaptatifs se composent d'un grand nombre d'agents qui interagissent par l'envoi et la réception de signaux. De plus, les

agents interagissent simultanément, en produisant un grand nombre de signaux simultanés. (ii) *Action conditionnelle*. Les actions des agents dans un système complexe adaptatif dépendent généralement des signaux qu'ils reçoivent. Autrement dit, les agents ont une structure contenant une série de règles (*SI signal x ALORS exécuter acte y*). L'acte peut lui-même déclencher un signal, qui peut être envoyé d'une manière ouverte dans l'environnement de l'agent, permettant des rétroactions assez compliquées. Ces séquences imbriquées de *règles-actions-signal,* qui, *de facto*, deviennent des programmes exécutés en parallèle, permettent une grande flexibilité ouvrant la voie à la possibilité d'exploration d'un grand éventail de possibilités. (iii) *Modularité*. Au niveau de l'agent, des groupes de règles se combinent souvent pour agir en tant que "sous-programmes". Par exemple, l'agent peut réagir à la situation actuelle en exécutant une séquence de règles. Ces "sous-programmes" agissent comme des blocs de construction qui peuvent être combinés pour gérer des situations nouvelles, plutôt que d'essayer d'anticiper chaque situation possible avec une règle distincte. Par le fait même que ces blocs de construction sont testés fréquemment, dans un large éventail de situations, leur utilité est rapidement confirmée ou infirmée. (iv) *L'adaptation* et *l'évolution*. Les agents d'un un système complexe adaptatif changent dans le temps. Ces changements sont généralement des adaptations qui améliorent la performance, plutôt que des variations aléatoires. L'adaptation exige la solution de deux problèmes : le problème de la répartition de l'erreur et le problème de la découverte de règle[39].

Les notions d'émergence et d'auto-organisation sont des notions importantes dans le domaine de la complexité. Un phénomène peut être qualifié d'émergent si l'application des outils analytiques traditionnels ne peut pas expliquer le comportement du système.

---

[39] De l'anglais (*the credit assignment problem and the rule discovery problem*)

Dans ces cas, le tout se comporte d'une façon qui ne peut pas être expliquée par l'étude des composants individuellement (c.à.d. agents) (Jones, 2003).

Différentes notions concernant le phénomène de l'émergence que les systèmes complexes adaptatifs peuvent exhiber, se côtoient dans la littérature et peuvent être utilisées parfois de manière interchangeable.

Les phénomènes émergents créés par les systèmes complexes, leurs capacités d'évolution et d'adaptation à leur environnement présentent des formes variées :

- L'auto-organisation concerne le processus par lequel se matérialise l'émergence de nouvelles formes ou de structures.
- L'évolution concerne la capacité de transformation des caractéristiques des composants du système (i.e. agents), de leurs dynamiques (i.e. règles de comportement). L'évolution peut être aussi apparentée au processus d'adaptation.
- La coévolution quant à elle concerne le cas où deux ou plusieurs composants (incluant environnement) s'affectent mutuellement avec leur évolution.

1.4.3 Modélisation et simulation à base d'agents

On trouve dans la littérature des termes similaires pour désigner l'approche qui consiste à modéliser et à simuler un système complexe : *modélisation et simulation à base d'agents*[40] (ABMS), *simulation multi-agents*[41] (MAS), *modélisation individus-*

[40] Dans cette thèse nous avons adopté le terme : ***modélisation et simulation à base d'agents*** « *Agent-based Modeling and Simulation* » (ABMS)

[41] De l'anglais « *Multi-agent simulation* »

centrée[42] (IBM), ***modélisation à base d'agents***[43] (ABM) ou *simulation à base d'agents*[44] (ABS) (Siegfried, 2014).

La *modélisation et simulation à base d'agents* est un nouveau paradigme de modélisation et un des plus excitants développement pratique dans la modélisation depuis l'invention des bases de données relationnelles (North et Macal, 2007). Bien que ce nouveau paradigme de modélisation soit lié à plusieurs domaines de recherche, notamment la théorie de la complexité, la théorie des systèmes, la dynamique des systèmes, l'informatique, la science de la gestion et les sciences sociales, il est cependant particulièrement lié aux approches de modélisation et de simulation classiques. Il s'appuie sur ces domaines non seulement pour ses fondements théoriques, sa vision conceptuelle et sa philosophie, mais aussi pour ses techniques de modélisation (Macal et North, 2009).

La modélisation à base d'agents et une modélisation ascendante « bottom-up », le système modélisé étant décrit comme une collection d'agents et sa dynamique globale comme le produit de ces interactions (Bonabeau, 2002).

1.4.3.1 La simulation

La simulation est considérée aujourd'hui, avec la théorie et l'expérimentation, comme le troisième pilier de la science. L'analyse de nombreux systèmes, processus et phénomènes est souvent faisable en développant et en exécutant des modèles de simulation (Minar *et al.*, 1996 ; Siegfried, 2014).

---

[42] De l'anglais « *Individual-based modeling* »

[43] De l'anglais « *Agent-based Modeling* »

[44] De l'anglais « *Agent-based Simulation* »

En science et spécialement dans l'étude des systèmes complexes, les programmes informatiques commencent à jouer un rôle important comme équipement scientifique (Minar *et al.*, 1996). Les simulations informatiques – logiciels comme outils expérimentaux - ont pris leurs places à côté des outils expérimentaux physiques. Les modèles informatiques offrent plusieurs avantages, par rapport aux méthodes expérimentales traditionnelles. Ils ont aussi plusieurs désavantages, en particulier la complexité de la tâche de la programmation du logiciel.

La simulation informatique est une discipline fondamentale de l'étude des systèmes complexes (Fishwick, s.d.).

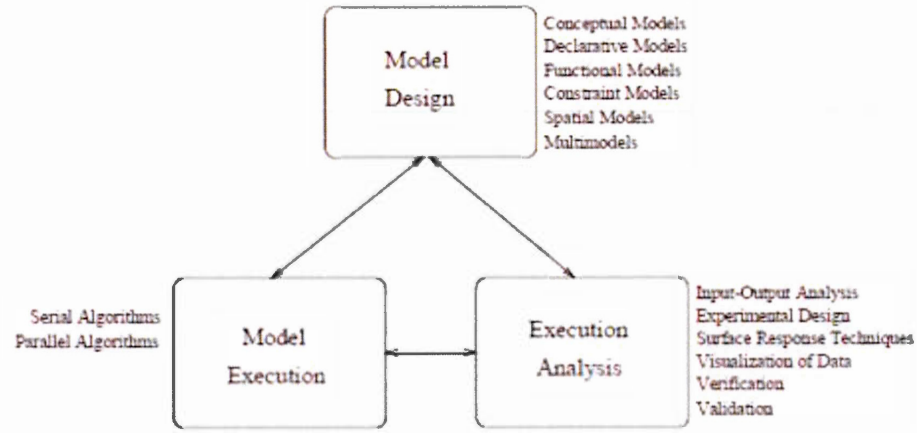


Figure 1.12 Taxonomie de la simulation **(Fishwick, s.d.)**

Le tableau 1.2 présente une classification (Treuil *et al.*, 2008), en cinq catégories, des différents usages et objectifs de la démarche de simulation. Ces objectifs peuvent naturellement être combinés.

Utilisations de la simulation	
Validation, Évaluation, Vérification	La simulation a comme objectif de tester une hypothèse du modèle du système de référence, de le vérifier ou d'accréditer la théorie qui a servi à le construire.
Communication, Formation, Visualisation	La simulation a comme objectif de « montrer » et de partager le modèle de la dynamique du système de référence.
Compréhension, Exploration, Explicitation	La simulation sert à comprendre le fonctionnement du système de référence, en considérant le modèle comme une réplique miniature qui pourra être étudiée plus facilement.
Contrôle, Action, Pilotage	La simulation a comme objectif de servir de support à une prise de décision ou à un contrôle qui influera sur l'état (réel) du système de référence
Prévision, Prédiction, Anticipation	La simulation sert à prévoir les évolutions possibles du système de référence en fonction d'évolution ou de perturbations spécifiques.

Tableau 1.2 Objectifs finaux de la simulation **(Treuil *et al.*, 2008)**

En règle générale, pour faire exécuter des simulations on doit avoir recours à des simulateurs. Un simulateur est un programme (ou plateforme) informatique capable d'interpréter des modèles dynamiques. Il est utilisé pour produire les perturbations désirées sur ces modèles (Treuil *et al.*, 2008).

1.4.3.2 La modélisation

Un modèle à base d'agents a typiquement trois éléments (Macal et North, 2010):

- Les agents : un ensemble d'agents exhibant des attributs et des comportements.
- Les relations : un ensemble de relations entre les agents basés sur des méthodes d'interactions. La topologie sous-jacente définie comment et avec qui l'agent interagit.
- L'environnement : les agents interagissent dans le contexte d'un environnement.

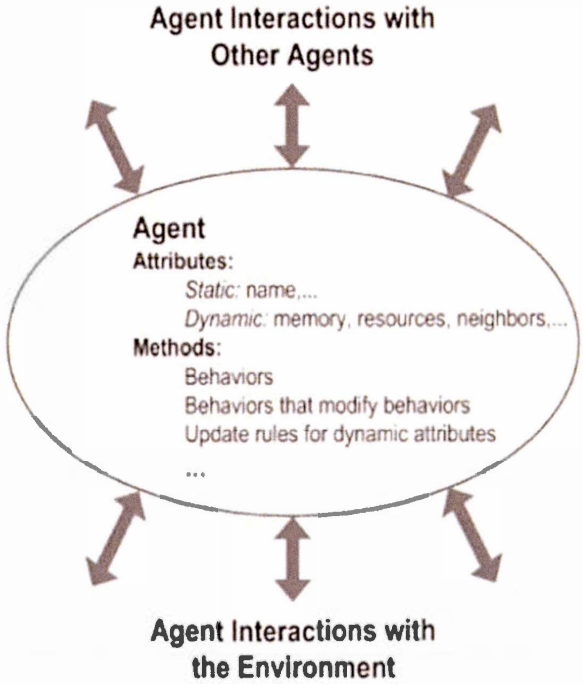


Figure 1.13 Structure d'un agent **(Macal et North, 2010)**

Russell et al. (Russell *et al.*, 2010) catégorisent la structure d'un agent en cinq types de base, par ordre de généralité grandissante :

- Agent réflexif simple : c'est le type d'agent le plus simple. Il se base sur des règles simples (i.e. *Si condition Alors action*). Il n'a pas de mémoire et ne réagit, donc, pas par rapport à ses états ou expériences passées.
- Agent réflexif avec état interne : ce type d'agent ressemble au précèdent dans la mesure où les deux se basent sur des règles simples type (i.e. *Si condition Alors action*). En revanche, ce type a besoin de maintenir un modèle décrivant son état actuel. Ce modèle va être à la base de la prise de décision.
- Agent basé sur les buts : Ce type se base sur les buts (objectifs) pour prendre des décisions sur ses actions. En effet, ce type d'agent ne s'appuie pas sur des règles simples, contrairement aux deux types précédents, pour décider de

l'action à prendre, mais engage des tâches d'exploration et de planification pour la recherche et la sélection d'actions lui permettant d'atteindre ses buts.

- Agent basé sur l'utilité : ce type est considéré comme étant le type le plus évolué dans la mesure où il raisonne sur les décisions concernant ses actions par rapport à leur utilité. En effet, l'objectif pour ce type d'agent est d'explorer et de planifier ses actions, tout comme le fait le type précèdent, mais en choisissant l'action qui va mener l'agent à un état avec la meilleur utilité (i.e. mesure de performance permettant le choix précis entre les états selon la satisfaction de l'agent).
- Agent apprenant[45] : ce type est capable d'opérer, avec des connaissances limitées, dans des environnements inconnus et devenir ensuite, au fil de ses expériences et de son apprentissage, capable de prendre des décisions lui permettant au final d'atteindre ses objectifs.

Il est à noter que l'apprentissage peut être considéré comme une faculté supplémentaire que l'agent peut avoir, quel que soit sont type.

Les agents communiquent selon deux modes de communication (Chaib-draa et Dignum, 2002):

- Direct : communication correspondant à un échange de messages, généralement structuré selon un protocole ou langage de communication entre agents. Ces langages permettent de représenter des actions ou des actes communicatifs (c.-à-d. Informer, Proposer, Accepter, etc.). Parmi ces langages, on trouve le langage FIPA-ACL[46] « *FIPA-Agent Communication Language* » ou encore le langage *KQML* « *Knowledge Query and Manipulation Language* ».

[45] De l'anglais « *learning agent* »

[46] Une initiative de la FIPA « *Foundation of Intelligent Physical Agents* ».

- Indirect : communication correspondant à un échange de messages fait par l'intermédiaire de l'environnement, plus précisément en générant un évènement à la suite d'actions effectuant des changements observables dans l'environnement. Par exemple, un agent qui verrouille une ressource, envoi aux autres agents, à travers de cette action, le message de son besoin de cette ressource à ce moment. Ce type de communication participe, en générale, à des interactions de type réactif (c.-à-d. interactions entre agents réactifs).

Par ailleurs, dans le contexte d'un environnement émergent et auto-organisé, la communication indirecte qui se matérialise avec la modification des agents et de leur environnement s'appelle *stigmergie.* La *théorie de la stigmergie* (Grassé, 1967) est une théorie qui s'intéresse à expliquer le mécanisme entièrement automatique derrière la coordination des actions individuelles dans l'accomplissement d'un travail collectif. Cette notion est une notion très importante pour les systèmes complexes adaptatifs, d'où la pertinence de préciser sa définition (Wikipédia, 2015): « *La stigmergie est un mécanisme de coordination indirecte entre les agents. Le principe est que la trace laissée dans l'environnement par l'action initiale stimule une action suivante, par le même agent ou un agent différent. De cette façon, les actions successives ont tendance à se renforcer et ainsi conduisant à l'émergence spontanée d'activités cohérentes, apparemment systématiques.* ».

La validation des modèles de simulation à base d'agents est un autre sujet qui prend de plus en plus d'importance dans la littérature concernant le domaine de *la modélisation et de la simulation à base d'agents*. On distingue trois types de validation (Remondino et Correndo, 2006):

- Validation empirique : cette validation se base sur la comparaison des résultats obtenus du modèle, par rapport à ce que l'on peut observer sur le vrai système.
- Validation prédictive : cette validation cherche à prouver que les résultats obtenus du modèle peuvent avoir une validité dans des situations qui ne sont

pas encore observables directement dans la réalité. Ce type de validation est souvent utilisé dans l'analyse « *what-if* ».

- Validation structurelle : cette validation cherche à savoir comment les résultats sont obtenus. Elle permet de s'assurer que le modèle calque bien la réalité.

Comme on va le voir en détail dans le chapitre 4, la validation empirique est le type de validation que l'on a adopté pour la validation de notre modèle de simulation à base d'agents pour l'appariement automatique de schémas.

1.4.4 Modélisation de la prise de décision sous l'incertitude

Durant un processus de raisonnement pour une prise de décision, l'incertitude, qui peut découler généralement d'un manque d'informations ou d'informations imparfaites, s'invite souvent comme une composante inévitable avec laquelle il faut obligatoirement composer pour arriver à prendre une décision. En effet, l'incertitude se retrouve souvent à l'origine des principales difficultés rencontrées lors des tâches de raisonnement et de décision.

Parmi les approches permettant la modélisation des connaissances incertaines, et ultimement la quantification de cette incertitude, on retrouve une approche issue de la théorie bayésienne à savoir *les réseaux bayésiens*[47], *réseaux probabilistes* ou encore *réseaux de croyances*[48]. Le fondement théorique de cette approche repose, en effet, sur *la théorie bayésienne*[49] (également appelée *le théorème*, *la loi* ou *la règle de Bayes*) publiée par le mathématicien Thomas Bayes en 1763 (Stutz et Cheeseman, 1994).

---

[47] De l'anglais « *Bayesian Networks* »

[48] De l'anglais « *Belief Networks* »

[49] De l'anglais « *Bayesian Theory* »

Un réseau bayésien est un graphe acyclique dirigé[50] dans lequel les nœuds représentent des variables aléatoires (par exemple, l'âge d'un patient) et les liens représentent les dépendances d'information ou de causalité entre les variables (Pearl, 2011). L'approche bayésienne décrit la probabilité d'un événement, en fonction des conditions qui pourraient être liées à l'événement. Par exemple, si un lien existe entre l'apparition d'une certaine maladie et l'âge des patients, alors, en utilisant le théorème de Bayes, on peut évaluer précisément la probabilité qu'un patient puisse développer cette maladie connaissant son âge.

L'équation mathématique décrivant la règle de bayes est la suivante :

$$P(A|B_1, B_2, \dots, B_n) = \frac{P(B_1, \dots, B_n|A) * P(A)}{P(B_1, \dots, B_n)}$$

(1.1) La règle de Bayes

Où :

- A et B sont des évènements et $P(B) \neq 0$
- $P(A)$ et $P(B)$ sont les probabilités que les évènements A et B soient observés séparément
- $P(A|B)$ représente la probabilité conditionnelle d'observer l'évènement A lorsque B est vrai
- $P(B|A)$ représente la probabilité conditionnelle d'observer l'évènement B lorsque A est vrai

Les ***diagrammes d'influence*** sont une extension aux *réseaux bayésiens*, proposée par Howard et al., permettant une représentation graphique compacte du problème de la prise de décision probabiliste. Les auteurs décrivent le diagramme d'influence comme

[50] De l'anglais « *Directed Acyclic Graph (DAG)* »

une façon de décrire les dépendances antre les variables aléatoires et les décisions : "*An influence diagram is a way of describing the dependencies among aleatory variables and decisions.*" (Howard et Matheson, 2005).

Outre le volet graphique, qui consiste à représenter le problème étudié par un graphe acyclique dirigé, tout comme les réseaux bayésiens, le diagramme d'influence, se base en plus sur un autre volet quantitatif qui concerne la quantification de l'incertitude relative aux relations d'influences (entre les variables) par le biais de tables de probabilités conditionnelles (CPT)[51].

Les *diagrammes d'influence* permettent aussi la possibilité d'associer une utilité à chaque décision possible dans le but d'avoir une analyse et une évaluation de la qualité du résultat du processus probabiliste de la prise de décision. En effet, les *diagrammes d'influence* sont fondamentalement une combinaison d'inférence bayésienne et d'estimation de l'utilité, basée sur une séquence de preuves et d'actions.

Les diagrammes d'influence comprennent les éléments suivants (Matsumoto *et al.*, 2011):

- Les nœuds de chance (variables aléatoires) : ce sont des variables aléatoires représentant des informations incertaines. Ces nœuds reflètent les incertitudes liées au problème de décision (représentés par des cercles).
- Les nœuds de décision : nœuds représentant un ensemble de choix possibles d'actions (représentés par des rectangles).
- Les nœuds d'utilité (valeurs) : ce sont des nœuds sans enfants représentant des fonctions d'utilité locales (degrés de satisfaction). Ces fonctions d'utilité représentent le gain (ou la perte, si négatifs) qui peuvent être obtenus à partir

[51] De l'anglais « *Conditional Probability Table* »

d'un ensemble particulier d'actions ou de preuves (représentés par des losanges).

- Les arcs informationnels (Influences informationnelles) : flèches menant à un nœud de décision (impliquant une précédence temporelle), qui montrent quelles variables doivent être connues par les décideurs au moment où la décision est prise.
- Les arcs conditionnels (Influences conditionnelles) : flèches conduisant à des nœuds de chance ou des nœuds d'utilité, qui représentent respectivement des dépendances de probabilité conditionnelles ou des entrées d'une fonction d'utilité.

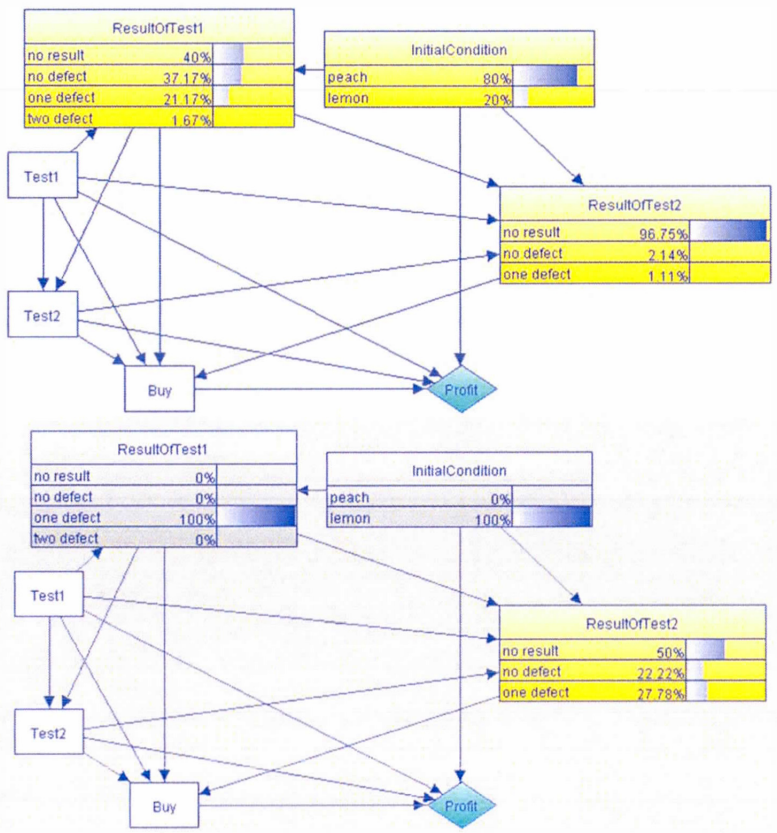


Figure 1.14 Exemple de diagramme d'influence **(Matsumoto *et al.*, 2011)**

La figure 1.14 illustre un diagramme d'influence pour un problème classique connu dans la littérature sous le nom de « *The Used Car Buyer* » (Howard, 1984). Cet exemple représente une situation où une personne veut décider d'acheter une certaine voiture d'occasion qui peut se révéler, apostériori, un bon ou un mauvais achat. *Test1*, *Test2* et *Buy* sont les nœuds de décision. Les boîtes plus grandes avec les barres de croyance sont les nœuds de chance (ou de probabilité). Le losange dans le coin inférieur droit est un nœud d'utilité.

Dans cette exemple, l'acheteur a la possibilité de faire (ou de ne pas faire) certains tests à l'avance (*Test1* et *Test2*), à la voiture d'occasion, et de l'acheter avec une garantie ou non (*Acheter*). Le nœud utilitaire (*Profit*) spécifie une fonction d'utilité qui indique le gain (ou la perte) estimé en fonction de la valeur de *InitialCondition* et des trois actions précédentes. L'objectif principal lors de l'évaluation (c'est-à-dire le raisonnement) est de déterminer une séquence d'actions qui maximisent la valeur attendue de cette fonction d'utilité.

Concernant ce problème de décision (achat de la voiture d'occasion), le *diagramme d'influence* permet d'identifier les meilleures actions à prendre (celles ayant la plus grande utilité) tel que faire passer un test à la voiture d'occasion et l'acheter avec une garantie.

1.5 Conclusion

Dans la première partie de ce chapitre, en plus d'introduire la problématique de l'automatisation de l'appariement et de montrer les avancées de la recherche dans ce domaine, nous avons essayé de mettre en relief les défis et difficultés qui freinent la vision de son automatisation complète. Plus particulièrement, on a essayé de se concentrer sur deux thèmes majeurs, en lien avec ces difficultés, à savoir : (i) le défis de l'incertitude qui est relié à la qualité du résultat de ce processus en l'occurrence l'alignement (c.-à-d. l'ensemble des correspondances) (ii) ainsi que les défis de la complexité caractérisant le processus de l'appariement et son l'optimisation.

Dans la deuxième partie de ce chapitre, mettre relief le mode de pense des approches existantes, à savoir le *réductionnisme*, et ensuite faire ressortir, les caractéristiques fondamentales et intrinsèques aux systèmes d'appariement existants (i.e. *la linéarité, la centralisation)* découlant de ce mode pensée, et ensuite discuté les corrélations qui peuvent exister entre ces caractéristiques et l'incapacité des systèmes à répondre complètement aux défis soulevés dans la littérature notamment la complexité et l'incertitude.

Dans la dernière partie de ce chapitre, on a présente une revue de littérature, non exhaustive, de la théorie des *système complexes* et plus particulièrement le paradigme des *systèmes complexe adaptatifs* qui se veut être notre cadre théorique pour notre approche systémique et holistique basée sur un modèle de simulation à base d'agents, pour l'appariement automatique de schémas.

CHAPITRE II

APPARIEMENT DES SCHÉMAS COMME UN SYSTÈME COMPLEXE ADAPTATIF

2.1 Introduction

Dans la première partie de ce chapitre, nous présentons de façon détaillée le cadre théorique dans lequel on se place pour aborder la problématique de l'appariement automatique de schémas, ainsi que notre cheminement méthodologique. Dès lors, dans un premier temps, on va s'atteler, à l'one de la théorie des systèmes complexes adaptatifs, à recentrer le problématique de l'appariement automatique de schémas, de sorte à clarifier et arrimer certains concepts de notre problématique de recherche au cadre théorique et vice versa. Dans un deuxième temps, on expose notre démarche méthodologique qui s'appuie principalement sur l'approche de modélisation et de simulation à base d'agents.

Dans la seconde partie, on propose une formalisation du modèle conceptuel de notre approche multidisciplinaire de simulation multi-agents pour l'appariement automatique de schémas. Aussi, on aborde sommairement notre vision de la transformation du modèle conceptuel en un programme informatique s'exécutant sur une plateforme de simulation.

On termine ce chapitre par une synthèse dont le but est de reprendre les principaux points abordés dans ce chapitre.

2.2 Cadre théorique

La théorie de la complexité et plus précisément la théorie des systèmes complexes adaptatifs représente en effet le socle théorique sur lequel repose notre approche pour l'appariement automatique de schémas. Dans un souci de simplification, on réfère, dans le cadre de cette thèse, à la théorie des système complexes adaptatifs comme la théorie englobant l'ensemble des concepts fondamentaux qui sont à la base de notre façon d'aborder et de concevoir le problème de l'appariement, notamment la systémique, la cybernétique, etc.

D'un autre côté, on réfère par extension au paradigme de modélisation et de simulation à base d'agents, comme l'ensemble des techniques permettant la modélisation, l'implantation et la simulation des systèmes complexes adaptatifs. Rappelons à ce propos, que la théorie des systèmes complexes adaptatifs se retrouve au centre même des fondements théoriques, de la vision conceptuelle et de la philosophie, de l'approche de modélisation et de simulation multi-agents (Macal et North, 2009).

On pense qu'à l'instar d'autres disciplines (sciences sociales, biologie, etc.), qui se sont appuyées sur la théorie des systèmes complexes adaptatifs, et qui ont pu exploiter avec succès ses caractéristiques fondamentales, l'appariement automatique de schémas est un domaine qui pourrait aussi se conceptualiser comme un système complexe adaptatif et ainsi tirer profit des caractéristiques fondamentales de cette théorie ainsi que de son approche de modélisation (i.e. modélisation et simulation à base d'agents), et ce dans le but d'apporter des réponses novatrices aux questions encore ouvertes dans ce domaine (e.g. incertitude).

Nous concédons que cette théorie des systèmes complexes adaptatifs pourrait sembler, de prime abord, comme inadaptée à la problématique de l'appariement automatique de schémas. Donc, et avant d'aller plus loin, essayons tout d'abord d'expliciter notre vision de la problématique de l'appariement automatique de schémas sous le prisme de la théorie des systèmes complexes adaptatifs.

En tout premier lieu, nous nous s'interrogeons sur la nature même des solutions d'appariement automatique de schémas, à savoir si elles peuvent être considérées comme des systèmes. La réponse est sans équivoque positive. Même si l'on se tient strictement et simplement à la définition du terme *système*, du latin *systêma,* qui désigne « un ensemble cohérent de parties étroitement liées », il est clair que l'appariement automatique de schémas est un système, dans la mesure où l'appariement automatique de schémas est un ensemble cohérent, constitué de plusieurs composantes, liées entre elles en l'occurrence les schémas, les apparieurs, etc. Maintenant, si l'on en vient au sens *systémique* (Bertalanffy, 1968) du terme, l'appariement automatique de schémas est un système dont les différentes composantes, par exemple les éléments de schémas (pouvant être représentés par des agents cherchant à trouver le meilleur appariement), ou encore les apparieurs (eux-mêmes pouvant être représentés par des agents de différents types : apparieurs textuels, structurels, etc.), doivent rentrer en interaction au sein de ce système avec l'objectif de produire un résultat final : *l'alignement*. La figure 2.1 représente les différents éléments qui peuvent composer un système d'appariement de schémas. Comme on peut le voir, ce système regroupe plusieurs éléments qui s'influencent les uns les autres où chaque changement au niveau d'un élément peut influencer le résultat final (l'effet papillon[52]).

[52] De l'anglais (*butterfly effect*)

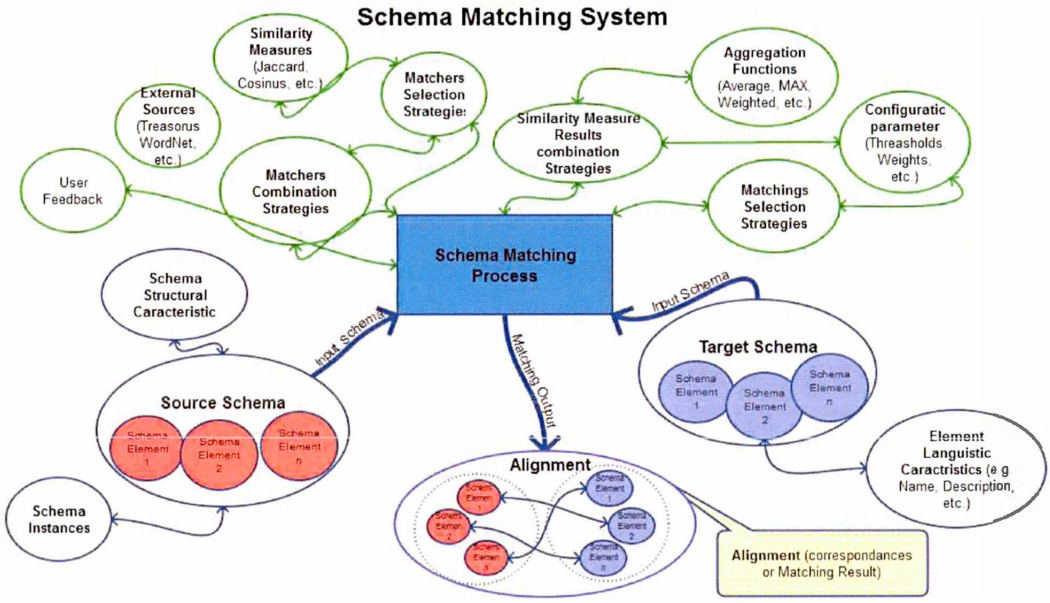


Figure 2.1 Le système d'appariement de schémas et ces composants

Ensuite, examinons pourquoi le système de l'appariement automatique de schémas peut être qualifié de complexe. Rappelons à cet effet, la distinction (Jones, 2003) entre un système « *complexe* » et un système « *compliqué* ». Premièrement, dans les systèmes complexes, les relations entre les agents (i.e. éléments du système) sont plus importantes que les agents eux-mêmes, contrairement aux systèmes compliqués où les éléments et leurs relations ont une importance équivalente. Deuxièmement, dans les systèmes complexes, des règles simples peuvent produire des réponses surprenantes et complexes à la fois, contrairement aux systèmes compliqués, où la résultante des algorithmes simples n'est autre que des réponses simples et prédictibles. Et enfin, dans les systèmes complexes les agents ont la latitude de répondre selon les limites des règles, par opposition aux systèmes compliqués, où la réponse des composants est complètement déterminée.

Partant de cette distinction, on considère que le processus d'appariement doit être pensé et modélisé comme un système complexe (notre approche) et non pas comme un

système compliqué (approches actuelles). Bien que dans la littérature, on désigne les approches actuelles de l'appariement automatique de schémas comme un processus, une tâche ou encore un système complexe, il reste que cette désignation fait simplement référence au terme « complexité » qui est utilisé, généralement par abus de langage, pour désigner la notion de *complication* et non pas la notion de *complexité* du point de vue des systèmes complexes.

De par leur mode de pensée (i.e. *réductionniste*) et leur méthodologie de modélisation (i.e. *analytique*), les approches existantes, et ce le cas de le dire, ne laissent rien au « hasard ». En effet, elles construisent des systèmes d'appariement automatique en prenant le soin de fixer les éléments constituant de ce système (e.g. choix prédéterminé des appariеurs, des stratégies de combinaison des matrices de similarité) et ensuite en dictant à ce système, dans le détail, son mode de fonctionnement. Cela conduit à la situation actuelle dans ce domaine, c'est à dire des systèmes qui peuvent certes être parfois performants, mais qui restent des systèmes compliqués, difficilement configurables et optimisables et ayant beaucoup de difficulté à s'adapter aux changements (i.e. prévisibles et déterministes).

En revanche, on pense que si l'on essaye de déléguer le contrôle et la prise de décision, qui sont centralisés au niveau des systèmes d'appariement existants, à des entités autonomes (i.e. éléments de schémas), capables, par exemple, de pouvoir faire une sélection aléatoire d'appariеur pour le calcul de similarité, ou encore capable de décider du manière autonome et consensuelle de former des paires (i.e. meilleur appariement pour une paire d'éléments de schémas), on se retrouve alors avec un système exhibant une dynamique non-linaire, complexe et difficile à prévoir. Grâce à cette qualité de la complexité, ce système devient alors plus que juste la somme de ses constituants.

Enfin, et en dernier lieu, expliquons-en quoi et comment l'appariement automatique de schémas serait adaptatif. Ce dernier point nous renvoie à la notion d'adaptation des systèmes complexes. En effet, face à des environnements sources d'imprévus, le

mécanisme d'adaptation (dans un système complexe), se traduisant par des changements dans le temps, donne aux agents la capacité de survivre, de croitre et de se maintenir dans cet environnement perturbé. Par voie de conséquence, on peut dire que ce mécanisme d'adaptation confère au système, dans son ensemble, stabilité et robustesse. Le mécanisme d'adaptation, selon (Holland, 2006), ne doit pas être vu seulement comme de simples variations aléatoires mais plutôt comme un mécanisme ayant pour objectif l'amélioration de la performance, et ce en passant par la résolution de deux problèmes principaux: le problème de la répartition de l'erreur et le problème de la découverte de règles[53]. Il serait pertinent de noter que la notion de l'adaptation est fortement reliée aux notions de rétroaction et d'homéostasie (Bonami *et al.*, 1993), dans le sens où le couplage de la rétroaction positive et négative conduit à un équilibre dynamique (homéostasie) grâce au processus d'adaptation, lequel peut être considéré comme un processus de régulation et d'innovation.

Concernant l'appariement automatique de schémas, on avance que grâce à des règles simples, les agents (i.e. les éléments de schémas) durant la simulation de notre modèle, peuvent montrer un haut niveau d'adaptation face à des situations complexes. Cette adaptation peut se manifester à plusieurs niveaux : (i) une adaptation aux perturbations liées aux changement des scénarios d'appariement (auto-configuration), ou encore (ii) une adaptation face aux modifications qui peuvent toucher l'assemblage du système d'appariement (i.e. déploiement de nouveaux composants), par exemple, l'ajout de nouveaux apparieurs ou le retrait d'apparieurs non performants (auto-configuration), et enfin (iii) une adaptation permettant l'exploration dynamique de l'espace des combinaisons possibles durant le processus d'appariement, dans le but de maximiser efficacement les paramètres et stratégies pour le processus d'appariement, et ce, sans pour autant subir le problème de l'explosion combinatoire (auto-optimisation). Notre vision de l'adaptation pour l'appariement automatique de schémas serait à notre avis

[53] De l'anglais (*the credit assignment problem and the rule discovery problem*)

la voie à suivre pour décharger l'utilisateur de la complexité et de l'effort lié à sa configuration et à son optimisation.

Il est indéniable que beaucoup de progrès ont été faits concernant la problématique de l'appariement automatique de schémas. Cependant, on demeure convaincu que les défis qui continuent à confronter cette problématique (notamment l'incertitude, l'optimisation, la configuration, etc.), requièrent, de notre point de vue, l'adoption d'une nouvelle vision *systémique* et *holistique* qui prendrait en considération l'ensemble des facteurs et paramètres influençant le système de l'appariement de schémas dans son entièreté. On pense que la concrétisation d'une modélisation de l'appariement de schémas comme un système complexe adaptatif, rendrait possible la conception d'un système d'appariement automatique de schémas, alliant simplicité et efficacité ce qui serait une contribution importante au domaine.

On pense que la conceptualisation de l'appariement de schémas sous la perspective des systèmes complexes adaptatives (c'est-à-dire en exploitant les caractéristiques fondamentales de ces systèmes) va permettre la conception d'un système affichant les qualités suivantes: (i) un système de compréhension facile, de par le fait que système serait décomposé en plusieurs entités simples « *agents* » (i.e. deux groupes d'agents représentant les schémas source et cible), interagissant selon des règles simples, et (ii) un système efficace et efficient, capable d'une manière autonome en évoluant dans le temps, de s'adapter, de s'auto-organiser et ainsi de faire émerger la solution pour n'importe quel scénario d'appariement. Étant donné que ce résultat émergeant (i.e. l'alignement ou les correspondances) émane des interactions, des choix et des comportements individuels des agents (basé sur des règles simples et éventuellement des capacités de raisonnement), cette efficacité visée, doit se traduire principalement par la réduction de l'incertitude sur l'alignement. L'efficience, quant à elle, doit se matérialiser par la réduction de l'effort nécessaire à cette efficacité notamment la réduction de l'effort utilisateur nécessaire à la configuration ou à l'optimisation du

système d'appariement, et cela sans nuire considérablement au temps de réponse qui devrait rester dans les limites du viable et de l'acceptable (surtout pour l'utilisation de l'appariement dans des environnements hautement dynamiques).

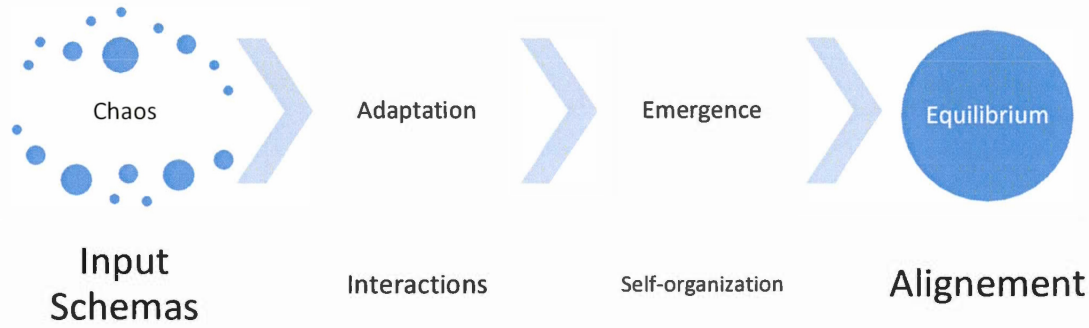


Figure 2.2 États du système complexe de l'appariement de schémas

La figure ci-dessus représente les différentes caractéristiques fondamentales permettant au système d'appariement de passer d'un état de *chaos* (i.e., état initial du système) à un état *d'équilibre* (i.e., état final du système). L'état initial du système est un état de *chaos* où les agents sont instables à cause de leur statut de l'appariement (appariement indéterminé). Après le début de la simulation, commence l'étape de l'adaptation où les agents rentrent en interaction à la recherche du meilleur appariement. Au fil des cycles de la simulation, des appariements consensuels (auto-organisation) commencent à se former (i.e., solution locale représentant un équilibre local à la paire d'agents), et ainsi faire émerger la solution finale de l'appariement à savoir l'alignement. En somme, l'état final du système est atteint une fois un équilibre pour le système dans son entièreté est trouvé signalant par le fait même la fin de la simulation.

Pour mieux situer note approche par rapport à la problématique de l'appariement automatique de schémas en général, et par rapport aux approches existantes en particulier, nous avons emprunté et adapté la *Matrice de Stacey* (Ralph, 2000). Cette matrice permet de replacer schématiquement, à l'aide d'un diagramme, certains

concepts de la théorie de la complexité, à savoir les concepts de simplicité, complication, complexité ou de chaos, et ainsi aide à mieux cerner le degré de complexité d'une situation ainsi que les actions à prendre, selon le niveau de certitude et le niveau d'accord, face à la situation en question.

On pense que si l'on adopte la *Matrice de Stacey*, pour situer les différentes approches concernant l'appariement automatique de schémas, toutes les approches existantes doivent être situées dans la zone de la complication dans la mesure où ces approches répondent à des besoins connus (e.g., optimisation ciblée pour une étape du processus d'appariement, des scénarios d'appariement pour un domaine d'affaire cible) avec une solution plus ou moins bien connue, ce qui soit dit en passant, explique la multiplication et la diversité des approches et des techniques utilisées.

En revanche, notre approche va se situer elle au niveau de la zone de complexité dans la mesure où notre approche a ceci de particulier, qu'elle est capable de s'adapter à de nouveaux besoins (e.g., déploiement d'un nouvel apparieur, changement de domaine d'affaire pour les scénarios d'appariement) et ceci en suivant des chemins inconnus lors de la conception du système (une solution est attendue sans prédéterminer comment le système va s'arranger pour la faire émerger).

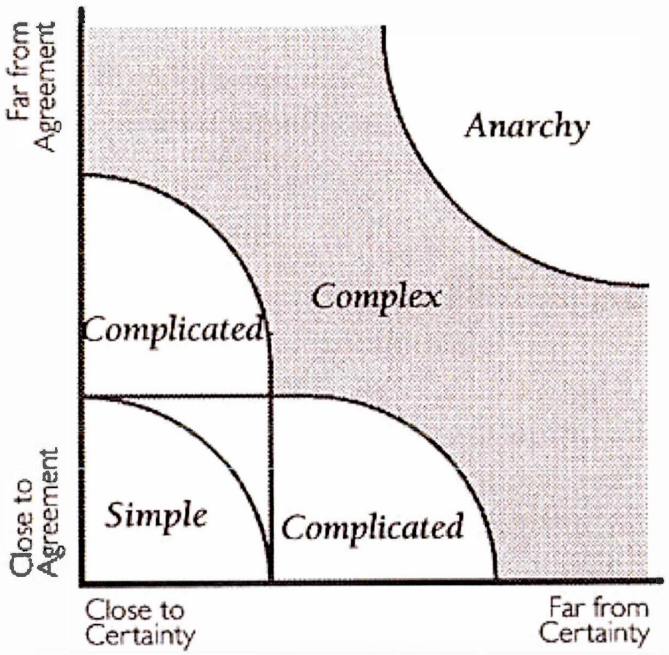


Figure 2.3 La Matrice de Stacey **(Zimmerman, 2001)**

Dans la zone de complexité de la *Matrice de Stacey*, notre solution trouve bien sa place pour gérer l'imprévisible. Pour être en mesure de gérer l'inconnu (i.e., changements, hétérogénéité et ambigüité), notre approche de simulation multi-agents et tout au long des interactions, est capable de faire émerger la solution finale par les mécanismes de coévolution (influence que peuvent exercer les agents les uns sur les autres et l'influence de nature stochastique que peut exercer l'environnement dans lequel évoluent les agents) et d'auto-organisation.

De toutes ces considérations, il serait opportun de rappeler l'hypothèse centrale de cette thèse, à savoir la viabilité de considérer l'appariement automatique de schémas comme un système complexe adaptatif, tirant bénéfice de la modélisation et simulation multi-agents.

2.3 Cheminement méthodologique

La démarche qui a été adoptée pour concrétiser notre approche, s'inspire de la démarche proposée par Shannon (Shannon, 1998). En effet, les étapes qui nous ont permis l'exploration, la conceptualisation, la conception et l'implémentation de notre modèle d'appariement automatique de schémas basé sur la théorie des systèmes complexes adaptatifs, peuvent être déclinées comme suit :

1. Définition du problème : lors de cette étape on a commencé par l'exploration de la problématique d'appariement de schémas sous l'angle d'un système complexe adaptatif :
    a. Faire une revue de littérature sur le domaine de l'appariement de schémas :
        i. Étudier et évaluer quelques outils existants tels que *COMA* (Do et Rahm, 2002), *Similarity Flood* (Melnik *et al.*, 2002), *YAM* (Duchateau *et al.*, 2009), etc. (dont le code est accessible sur Internet)
    b. Faire une revue de littérature sur le domaine des systèmes complexes adaptatifs et de la modélisation et simulation à base d'agents
        i. Effectuer des expérimentations sur quelques plateformes de simulation, à savoir *NetLogo* (Tisue et Wilensky, 2004) et *Repast Simphony* (North *et al.*, 2007)
        ii. Tester les idées préliminaires, touchant à la conceptualisation de l'appariement automatique de schémas (i.e. modélisation des éléments de schémas comme agents), avec l'outil *NetLogo*
        iii. Étudier des domaines connexes comme l'intelligence artificielle, incluant l'apprentissage machine, la théorie bayésienne (e.g., la prise de décision sous incertitude), etc.

2. Formulation du modèle conceptuel (Chapitre 2) : proposer une formalisation d'un modèle conceptuel à base agents pour le processus d'appariement qui soit agnostique à l'implémentation (i.e., engin de simulation) :
    i. Proposer les concepts fondamentaux sur lesquels s'articule le modèle conceptuel : (i) l'agent élément de schéma, (ii) la stochasticité et l'adaptation, (iii) l'appariement consensuel, (iv) l'auto-organisation et l'émergence, et enfin (v) les simulations multiples et l'analyse statistique.
3. Transcription du modèle (chapitres 3 et 5) : transformer ce modèle conceptuel en un programme informatique s'exécutant sur une plateforme de simulation
    a. Proposer une première version de notre prototype basée sur des agents de type réactif (chapitre 3) :
        i. Proposer une définition détaillée de l'agent réactif
        ii. Implémenter le prototype avec l'outil *Repast Simphony* (North *et al.*, 2007)
        iii. Valider le modèle sur un exemple de référence
        iv. Concevoir un apparieur prédictif basé sur l'apprentissage machine (classificateur bayésien) et ce, en exploitant la richesse des informations générées par les simulations
    b. Proposer une évolution (sur le plan conceptuel) de notre prototype, basée sur des agents de type cognitif/rationnel (chapitre 5) :
        i. Faire évoluer l'architecture interne à l'agent, d'un modèle conceptuel d'un agent de type réactif vers un agent de type cognitif/rationnel
        ii. Définir les nouveaux comportements de l'agent cognitif/rationnel, basés sur un engin de raisonnement pour la

sélection des actions (basé sur modèle bayésien de décision sous incertitude appelé, diagramme d'influence[54])

4. Expérimentation (chapitre 4) : évaluer d'une manière empirique, la viabilité de l'approche proposée et ce, en comparant ses performances aux résultats attendus :
    a. Développer différents scénarios d'appariement (avec les résultats attendus) dans le but de comparer les résultats générés par notre prototype avec les résultats attendus.
    b. Valider, confirmer ou infirmer les hypothèses avancées dans cette thèse :
        i. L'adéquation de la problématique d'appariement de schémas sous l'angle d'un système complexe adaptatif
        ii. L'augmentation de la qualité du résultat (efficacité)
        iii. La compréhension facile du système (Simplicité)
        iv. L'adaptation face aux changements (auto-configuration et auto-optimisation)

[54] De l'anglais « *Influence Diagram* »

Le figure 2.4 résume les principales étapes de notre cheminement de recherche.

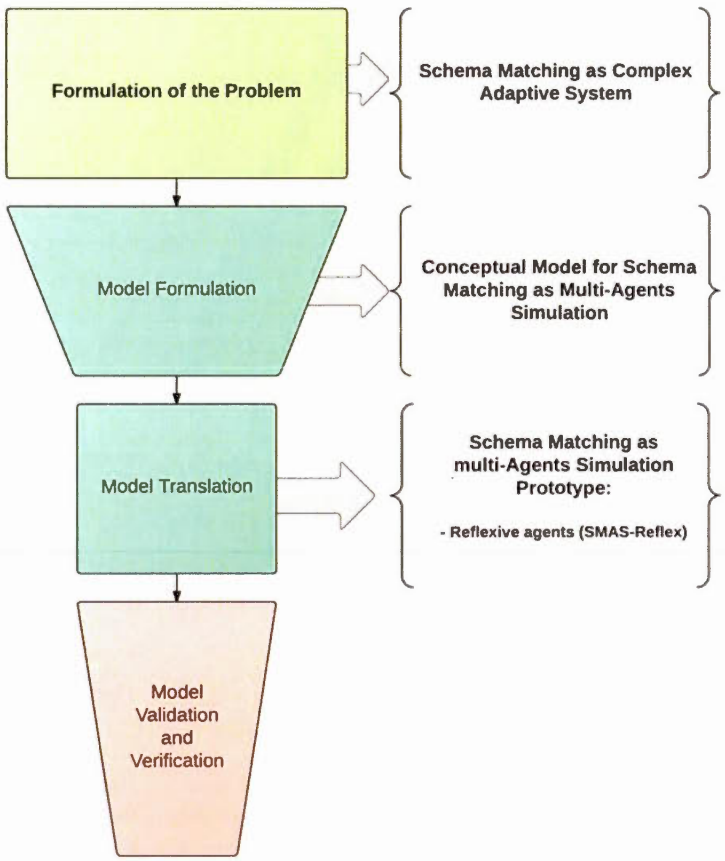


Figure 2.4 Cheminement méthodologique

Dans la deuxième partie de ce chapitre, nous présentons notre modèle conceptuel pour l'appariement automatique de schémas basé sur la modélisation et la simulation multi-agents.

2.4 Modélisation et simulation à base d'agents pour l'appariement de schémas (SMAS)[55]

2.4.1 Exemple de référence

Avant de présenter la formulation de notre modèle conceptuel, supportant l'idée de la modélisation de l'appariement automatique de schémas comme un système complexe adaptatif, et dans un souci de mieux expliquer les concepts fondamentaux rattachés à cette formulation, le chapitre courant va s'appuyer sur un exemple de référence d'un scénario d'appariement qui sera aussi repris tout au long du chapitre 3.

Supposons, que pour un scénario d'appariement donné, on dispose non seulement des deux schémas à apparier (schémas source et schéma cible), mais aussi de l'alignement attendu (l'ensemble des correspondances correctes produites par l'expert).

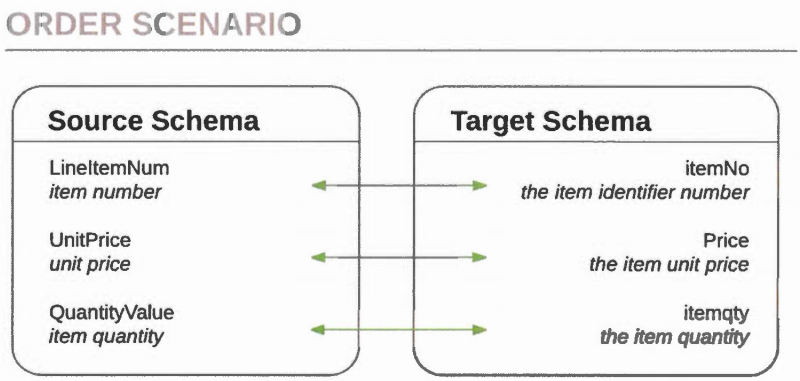


Figure 2.5 Exemple de référence pour l'appariement

[55] De l'anglais « *Schema Matching as Multi-Agents Simulation (SMAS)* ».

La figure 2.6 illustre un système générique d'appariement automatique (haut-niveau), permettant de produire le résultat attendu pour cet exemple de référence.

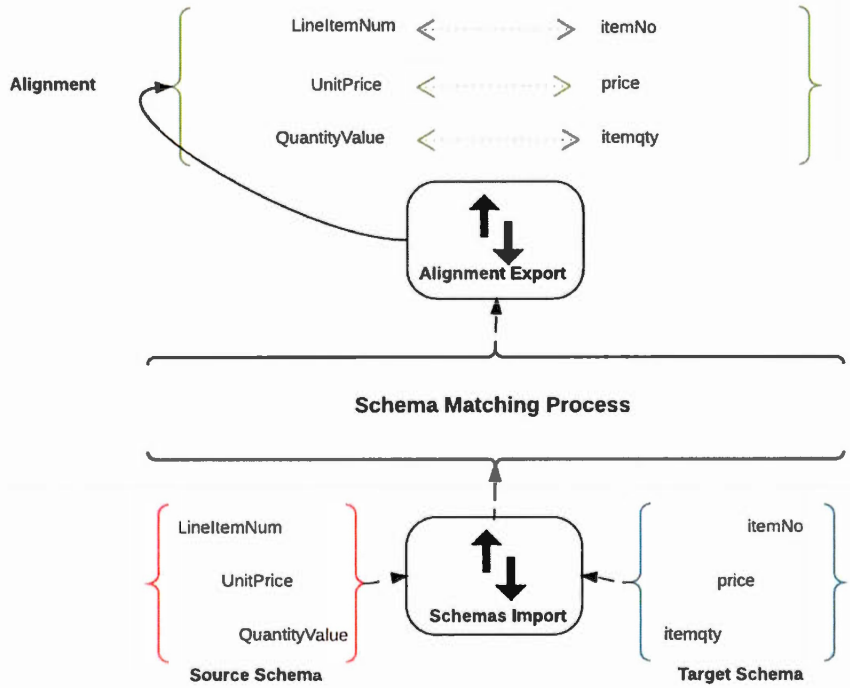


Figure 2.6 Processus d'appariement pour le scénario exemple de référence

Dans la prochaine section, nous formulons notre vision de la conceptualisation de l'appariement automatique de schémas comme un système complexe adaptatif. Cette conceptualisation consiste en une formalisation d'un modèle de simulation multi-agents, pour l'appariement automatique de schémas, qui soit agnostique à l'implémentation (i.e. engin de simulation).

2.4.2 Formulation du modèle conceptuel

Notre vision de la conception d'un système d'appariement place notre approche dans une classe à part par rapport aux approches existantes. Ces dernières, comme on l'a expliqué plutôt, même si elles peuvent sembler être différentes, partagent toutes les mêmes caractéristiques fondamentales découlant du même système de pensée réductionniste.

Notre modèle conceptuel, quant à lui, se place dans la lignée des approches issues de la pensée holistique ou systémique et naturellement se voit grandement influencé par les fondements théoriques découlant de la théorie des systèmes complexes adaptatifs.

Concernant le choix de la méthode de modélisation de notre modèle conceptuel, la modélisation et simulation à base d'agents a été pour nous un choix qui s'imposait de lui-même. À ce titre, rappelons une courte, mais un peu plus claire, mise en contexte sur le lien entre les systèmes complexes adaptatifs et la modélisation à base d'agents (Macal et North, 2010): « *Agent-based modelling is a way to model the dynamics of complex systems and complex adaptive systems.* ».

La matérialisation et l'exploitation des principes et caractéristiques derrière la théorie des systèmes complexes adaptatifs, notamment la non-linéarité, la stochasticité, l'auto-organisation et l'émergence, pour la conception d'une approche systémique de l'appariement de schémas, passe par l'utilisation de la modélisation et de la simulation à base d'agents.

L'idée centrale (et quelque peu intuitive) derrière notre modèle de simulation multi-agents pour l'appariement automatique de schémas, voudrait que l'on cesse de voir les schémas à apparier comme des intrants statiques au système d'appariement, mais plutôt, une fois importés dans le système, comme un ensemble d'agents représentant les éléments de schémas, divisé en deux groupes et où chaque groupe représente un schéma donné (schéma source ou schéma cible). Des lors, ces éléments de schémas

deviennent des agents autonomes capables de poursuivre un but, dans le cadre d'une simulation, à savoir trouver la meilleure correspondance dans le groupe opposé. La simulation des comportements de ces agents et de leurs interactions avec leur environnement, au niveau micro, fait émerger au niveau macro, un réseau auto-organisé[56] représentant la solution globale à l'appariement (i.e. relations entre les éléments des schémas). En d'autres termes, la résolution du processus d'appariement passe par l'effort individuel que déploie chaque agent d'un groupe (groupe source ou cible), d'une manière autonome et locale, tout au long de la simulation, pour trouver la meilleure relation de correspondance dans le groupe opposé (meilleur appariement[57]).

[56] De l'anglais « self-organized »

[57] De l'anglais « *Best Matching* »

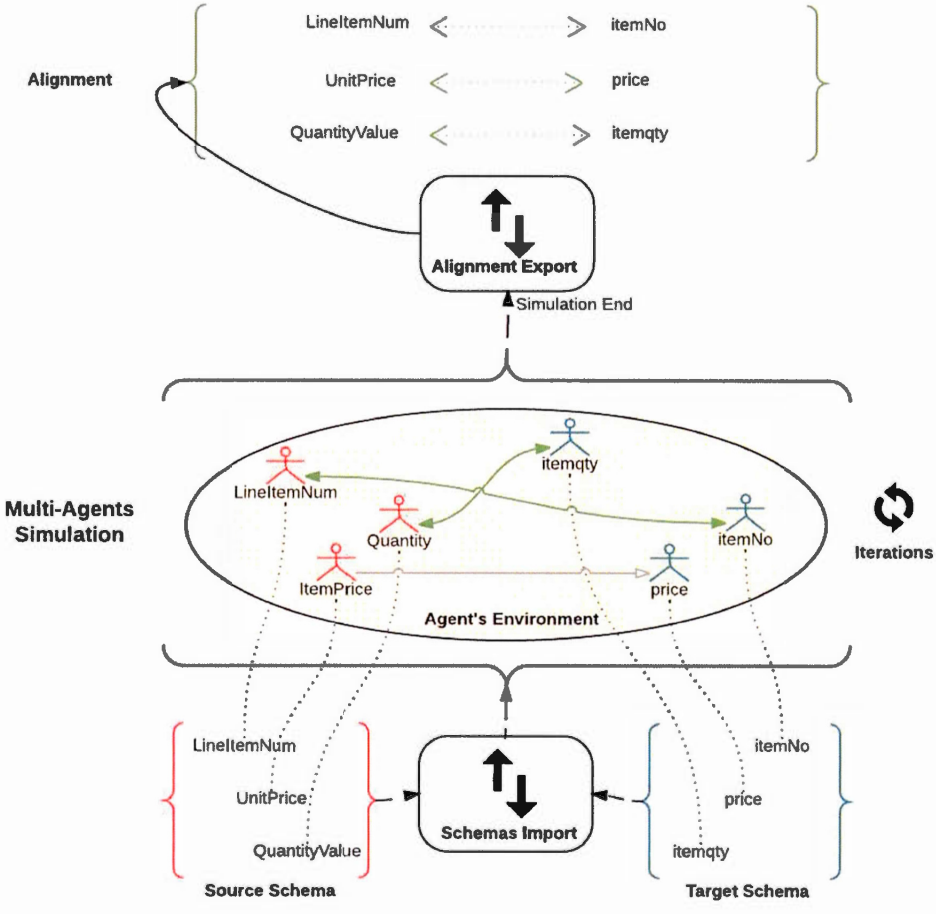


Figure 2.7 Simulation multi-agents pour l'appariement de schémas

D'une façon très schématique, la figure 2.7, met en perspective la place de la simulation multi-agents dans le processus général de l'appariement automatique de schémas.

Notre modèle conceptuel de simulation multi-agents pour l'appariement de schémas, s'articule autour de cinq concepts fondamentaux : (i) *l'agent élément de schéma* (*SE-Agent*), (ii) *la stochasticité* et *l'adaptation*, (iii) *l'appariement consensuel,* (iv) *l'auto-organisation et l'émergence de l'alignement*, et enfin (v) *les simulations multiples et l'analyse statistique* (quantification de l'incertitude).

2.4.2.1 Agent élément de schéma (*SE-Agent*)

Une des étapes clés dans notre cheminement vers la conceptualisation de l'appariement automatique de schémas en un système complexe adaptatif, a été sans aucun doute, l'*agentification*[58] de l'appariement automatique de schémas.

Cette *agentification* doit passer par une compréhension profonde et étendue du modèle derrière les systèmes d'appariement classiques, et plus particulièrement derrière la logique du contrôle centralisé des différentes étapes du processus (e.g. sélection et combinaison d'appariеurs, calcul de similarité et construction des matrices et/ou cubes de similarité, sélection des meilleures correspondances), ainsi que derrière la dynamique générale gouvernant ce processus. Cette compréhension est ensuite transposée vers un nouveau modèle décentralisé, qui consiste à déléguer le contrôle des étapes d'appariement à des entités autonomes avec une dynamique d'évolution et de coévolution, dans le temps, leur permettant l'atteinte de leur objectif premier, à savoir trouver le meilleur appariement dans le groupe opposé.

Ainsi donc, on peut dire que le composant central de notre modèle à base d'agents est *l'agent élément de schéma*[59] (*SE-Agent*). Ce dernier durant chaque itération de la simulation, passe par trois phases principales :

- Perception de l'environnement : *l'agent élément de schéma* est capable de percevoir son environnement de différentes manières (Il perçoit son environnement non pas à l'aide de capteurs proprement dit). D'un côté, il est capable d'interroger son environnement pour connaitre les agents disponibles pour appariement dans le groupe opposé. À la suite de quoi, il commence sa quête à la recherche du meilleur appariement et ce, en calculant le degré de

[58] De l'anglais « *Agentification* »

[59] De l'anglais « *Schema Element Agent* »

similarité entre lui et les autres agents disponibles pour l'appariement dans le groupe opposé. D'un autre côté, Il est capable de recevoir des messages de l'environnement (sous la forme d'évènements), lui notifiant qu'un agent du groupe opposé l'a désigné comme un appariement potentiel, ce que l'on appelle « *appariement candidat* », c'est-à-dire comme le meilleur agent, pour l'instant, pour former une relation de correspondance potentielle.

- Raisonnement sur l'action : durant cette phase, *l'agent élément de schéma*, et à la lumière des résultats obtenus lors de la phase de perception, raisonne sur l'action à prendre. Le raisonnement sur l'action est ce qui fait la différence entre les deux versions de notre prototype. En effet, la première version (i.e. *Reflex-SMAS*) est basée sur un agent de type réflexif dont la prise de décision est basée sur des règles simples ainsi que sur la réaction à des évènements. En revanche, la deuxième version de notre prototype (i.e. *Rational-SMAS*), dispose elle, de capacités de raisonnement plus évoluées dans la mesure où elle affiche, à l'aide d'un réseau de décision bayésien (i.e. diagramme d'influence), la possibilité de faire le choix entre des actions conflictuelles. Le choix de l'action une fois décidé, va conditionner ensuite la transition entre les différents états internes de l'agent.
- Action sur son environnement : là encore, *l'agent élément de schéma* est capable d'agir sur son environnement de différentes manières. Une fois, qu'il a trouvé un agent, du groupe opposé, comme le meilleur appariement pour lui, il le désigne alors comme un *appariement candidat* et demande à l'environnement de créer un arc unidirectionnel pointant vers cet agent (relation de correspondance candidate). Si l'autre agent du groupe opposé, à son tour, désigne l'agent comme un *appariement candidat* (se désignent mutuellement et en même temps comme *appariement candidat*), alors les deux « *agents éléments de schéma* » demandent à l'environnement de créer des arcs bidirectionnels pour signaler la création d'un appariement que l'on appelle « *appariement consensuel* » et ainsi passer à un état de stabilité.

Il est à noter que malgré l'interaction constante avec son environnement, notre agent en a une connaissance limitée, dans la mesure où il ne perçoit pas les choix et préférences internes des autres agents.

Le tableau ci-dessous résume les caractéristiques importantes de *l'agent élément de schéma.*

Agent Élément de Schéma (SE-Agent)	**Caractéristiques**
Attributs	Nom et commentaire
Buts	Chercher la stabilité en trouvant la meilleure correspondance dans le groupe opposé
Comportements	Calculer la similarité (nom et commentaire), sélectionner un appariement candidat et enfin chercher un appariement consensuel stable
Communication	Communication indirecte entre les agents. Cette communication se matérialise par la modification des agents de leur environnement (stigmergie)

Tableau 2.1 Caractéristiques de « *Agent Élément de Schéma* » (*SE-Agent*)

Ainsi que le met en évidence la figure 2.8, les agents ayant comme attributs un nom et une description (les attributs *name* et *comment* des éléments de schémas), et comme but ultime la recherche du meilleur appariement dans le groupe opposé, évoluent et interagissent dans un environnement dont la topologie sous-jacente est de type réseau. Cette topologie réseau permet d'identifier les interactions et les relations que les agents établissent entre eux et qui contribuent au final à l'émergence d'une structure auto-organisée représentant la solution finale du scénario d'appariement.

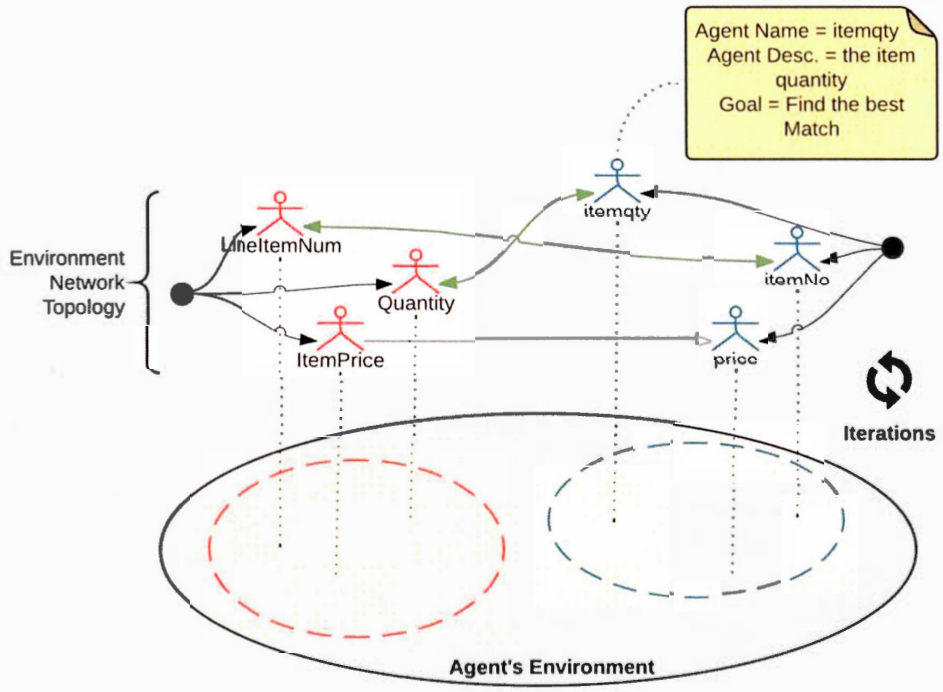


Figure 2.8 Représentation d'un « *agent élément de schéma* » (*SE-Agent*)

Précisons tout de même que dans un souci de simplicité, notre modèle conceptuel ne reprend pas forcement tous les aspects qui ont été modélisé dans les approches existantes, comme par exemple la structure des schémas ou encore les instances de schémas (i.e. données), non pas parce que notre modèle ne permet pas l'intégration de ces différents aspects ou présente une inadéquation avec ces aspects, mais bien au contraire. Si l'on prend par exemple, la structure du schéma, il aurait été tout à fait possible de l'exploiter en la modélisant comme une structure comprenant des regroupements d'agents au sein du même groupe, et ainsi reproduire la notion de *hiérarchisation des niveaux* qui fait partie de la nature même des systèmes complexes adaptatifs (groupes d'agents tels que départements dans une organisation, etc.). On est conscient que l'exploitation de la structure de schéma, ou encore l'exploitation des

données des instances de schémas, comme ça a été fait dans plusieurs travaux, aurait pu apporter des sources supplémentaires d'information intéressantes pour notre modèle. Cependant on a décidé de donner la priorité à la validation de notre hypothèse centrale (appariement de schémas comme système complexe adaptatifs) et garder ces idées, une fois la validation de l'hypothèse établie, pour des améliorations futures.

2.4.2.2 Stochasticité et adaptation

Compte tenu de l'aspect limitant du déterminisme et de la prévisibilité des solutions actuelles (les mêmes entrants conduisent aux mêmes extrants), notre modèle quant à lui a intégré la stochasticité comme une caractéristique intrinsèque pour surmonter ce déterminisme et cette prévisibilité. La stochasticité, en permettant l'injection de variations dans l'environnement des agents (i.e. bord du chaos), va forcer les agents à l'adaptation et ainsi stimuler l'émergence de solutions surprenantes et paradoxalement stables.

Ce paradoxe peut être expliqué par le principe de « ***l'ordre à partir du bruit***»[60] qui a été formulé par le cybernéticien[61] (Foerster, 2007) en 1960. Ce principe stipule que l'auto-organisation est facilitée par des perturbations aléatoires (i.e. du bruit), ce qui permet au système d'explorer un grand nombre d'états dans son espace d'états. En d'autres termes, un certain niveau de bruit dans un système auto-organisateur, lui évite de se figer et ainsi devenir inadaptable. D'autres théories ont présenté des principes similaires tel que « ***l'ordre via les fluctuations*** »[62]. Toutes ces théories tendent à statuer

---

[60] De l'anglais (*order from noice*)

[61] Wikipedia : « *La cybernétique (en anglais cybernetics) est un terme, formé à partir du grec κυβερνήτης (kubernêtês) « pilote, gouverneur », proposé en 1947 par le mathématicien américain Norbert Wiener pour promouvoir une vision unifiée des domaines naissants de l'automatique, de l'électronique et de la théorie mathématique de l'information, en tant que « théorie entière de la commande et de la communication, aussi bien chez l'animal que dans la machine ».*

[62] De l'anglais « *order through fluctuations*»

que l'auto-organisation a besoin d'entropie et de diversité pour que le système continue à s'adapter et à évoluer dans le temps. De fait, on peut s'interroger comment les systèmes d'appariement déterministes (approches existantes) peuvent prétendre à la faculté d'adaptation à la nouveauté, à la capacité d'accepter des perturbations et tolérer du bruit (e.g. changement des scénarios d'appariement, changement des configurations, déploiement de nouvelles composantes), et cela sans pour autant exploiter le principe de *l'ordre à partir du bruit* et sans avoir dans leur boite à outils, un outil comme la *stochasticité*. Selon *la théorie de la cybernétique* cela ne serait pas possible, car pour qu'un système puisse tolérer des aléas, pour qu'il puisse supporter la nouveauté et s'y adapter, une part d'indétermination et de *bruit* est nécessaire.

Notre modèle conceptuel, quant à lui, va se baser à différents moments sur différents niveaux de *bruit* et *d'aléatoire* (i.e. *stochasticité*), ce qui lui permet d'évoluer et de s'adapter et enfin faire émerger une auto-organisation. L'introduction du bruit et de l'aléatoire garanti un degré de liberté qui permet à notre modèle d'explorer une multitude de chemins et de possibilités lui permettant, *in fine*, d'atteindre son but.

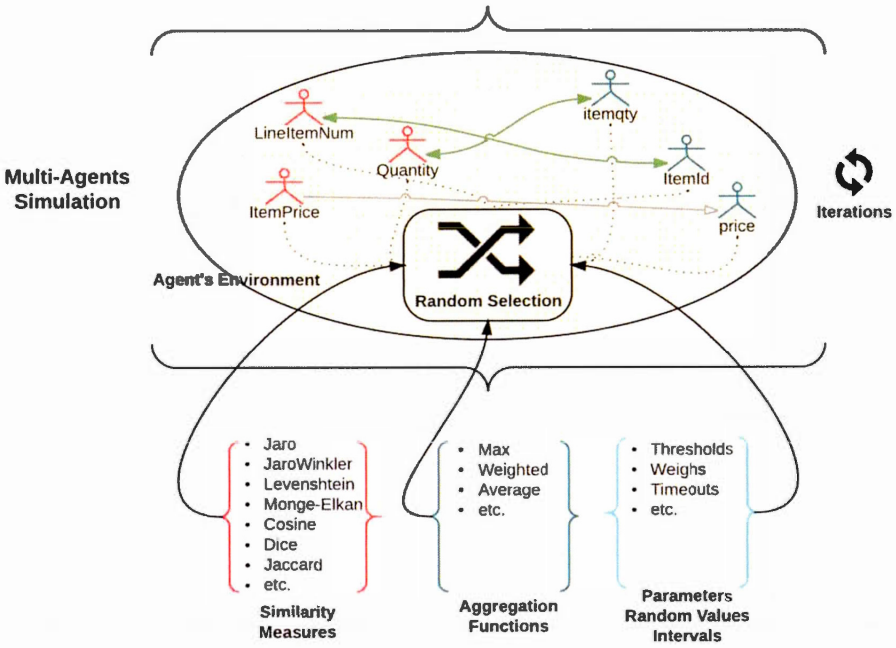


Figure 2.9 Appariement stochastique

La figure 2.9 décrit comment notre modèle fait usage de l'aléatoire. En fait, dans notre modèle, les agents, durant chaque itération de la simulation, font appel à l'environnement pour la sélection des mesures de similarité (e.g. *Jaro, JaroWinkler, Levenshtein*), des fonctions d'agrégation (i.e. *Max, Average, Weighted*) ou encore des différentes valeurs requises pour accomplir les différentes étapes de l'appariement (e.g. seuils, poids). À chaque fois que l'environnement est sollicité par un agent (lors de chaque itération), l'environnement retourne aux agents la composante ou la valeur demandée d'une manière totalement aléatoire. En d'autres mots, pour le calcul de similarité par exemple, l'agent n'a aucune connaissance préalable de la mesure de similarité qui il va utiliser durant l'itération, pour trouver l'agent le plus similaire dans le groupe opposé. Il en va de même pour les fonctions d'agrégation ou encore pour les autres paramètres.

Ce qui revient à dire qu'à chaque itération, l'agent aura l'occasion de tester de nouvelles combinaisons et cela sans tomber dans le problème de *l'explosion combinatoire* que peut engendrer la tentative d'une exploration exhaustive de l'espace de recherche des combinaisons possibles. On pense donc, que l'exploitation des notions de simulation et de la stochasticité dans notre modèle est la clé pour surmonter le problème de *l'explosion combinatoire*, dans le contexte de l'appariement de schémas, qui a été discuté dans plusieurs recherches (Rahm et Bernstein, 2001b) et qui continue de constituer un véritable frein face aux approches existantes.

2.4.2.3 Appariement consensuel

Pour un processus d'appariement générique, l'étape de la sélection des appariements prometteurs est une étape cruciale (l'étape de la *sélection des correspondances*[63]). Lors de cette étape les différentes stratégies qui peuvent être appliquées peuvent l'être appliquées selon différentes directions telles que notamment, unidirectionnelle, bidirectionnelle, ou encore selon l'algorithme du *mariage stable*[64] (issu de *la théorie des jeux*[65]) (Gale et Shapley, 1962). Cet algorithme a été associé dans la littérature à deux étapes de l'appariement à savoir la *combinaison des similarités*[66] ainsi que l'étape de la *sélection des correspondances*. En effet, (Rahm, 2011) décrivent le problème du mariage stable comme une solution raisonnable pour la *sélection des correspondances* (dont le degré de similarité excède typiquement un certain seuil), dans le sens où une correspondance est sélectionnée seulement si l'un des deux éléments de schémas constituant la paire est le plus similaire à l'autre élément de la paire et vice versa (réciprocité). Dans la même veine, l'algorithme du *mariage stable* est cité comme

---

[63] De l'anglais « *selection of correspondences* »

[64] De l'anglais « *stable Marriage* »

[65] De l'anglais « *game theory* »

[66] De l'anglais « *similarity combination* »

faisant partie des techniques permettant la déduction des correspondances (Bellahsene et Duchateau, 2011). (Marie et Gal, 2008) décrivent aussi l'algorithme du *mariage stable* comme moyen pour renforcer la contrainte de cardinalité 1:1.

L'algorithme du *mariage stable* est un algorithme efficace qui a fait ses preuves pour résoudre des problèmes nécessitant un appariement stable (e.g. appariement hôpitaux-patients ou encore appariement facultés de médecine-étudiants). Il a aussi souvent été utilisé par les approches classiques d'appariement. C'est pour ces raisons-là, que l'on avait décidé, au début de la conceptualisation de notre modèle, de se penché sur cet algorithme pour évaluer à quel point on pouvait à notre tour en tirer avantage. Surtout qu'avec *l'agentification* de notre modèle, cet algorithme semblait, à priori, être adéquat pour notre modèle.

Cependant, et après évaluation, on s'est rendu compte que cet algorithme affichait une centralisation, un déterminisme et une rigidité qui n'allait pas du tout dans le sens des caractéristiques de liberté de choix, d'aléatoire et de décentralisation qui singularisent notre modèle. Hormis la nature, centralisé et déterministe, de l'algorithme du *mariage stable*, cet algorithme favorise toujours la stabilité au détriment du degré de satisfaction des entités à apparier (les entités doivent se contenter, des fois, de choix qui n'étaient pas leurs premiers choix), ce qui va à l'encontre de l'objectif que notre modèle trace aux *agents éléments de schémas,* à savoir trouver le meilleur appariement dans le groupe opposé.

Dans notre modèle, on introduit donc une nouvelle approche pour la *sélection des correspondances* que l'on a baptisée *l'appariement basé sur le consensus*[67] (ou pour faire plus court *appariement consensuel*). Ce dernier, grâce aux propriétés de *l'évolution* et de la *coévolution* de notre modèle, et qui inscrivent la recherche et la sélection de l'appariement de la part de chaque agent dans une dimension temporelle,

[67] De l'anglais « *Consensus-based Matching* »

donne aux agents la possibilité (grâce à l'évolution dans le temps) de converger vers une solution localement optimale pour les deux agents (consensuelle), et ce même si les deux agents faisant partie du consensus peuvent emprunter des trajectoires totalement différentes pour arriver à ce consensus.

Ainsi que le met en lumière la figure 2.10, l'*appariement consensuel* signifie qu'aucune décision n'est prise sur la sélection d'une correspondance avant que les deux parties, chacun de son côté et d'une manière unilatérale, ne se désignent mutuellement comme meilleur appariement dans le groupe opposé (*appariement candidat*).

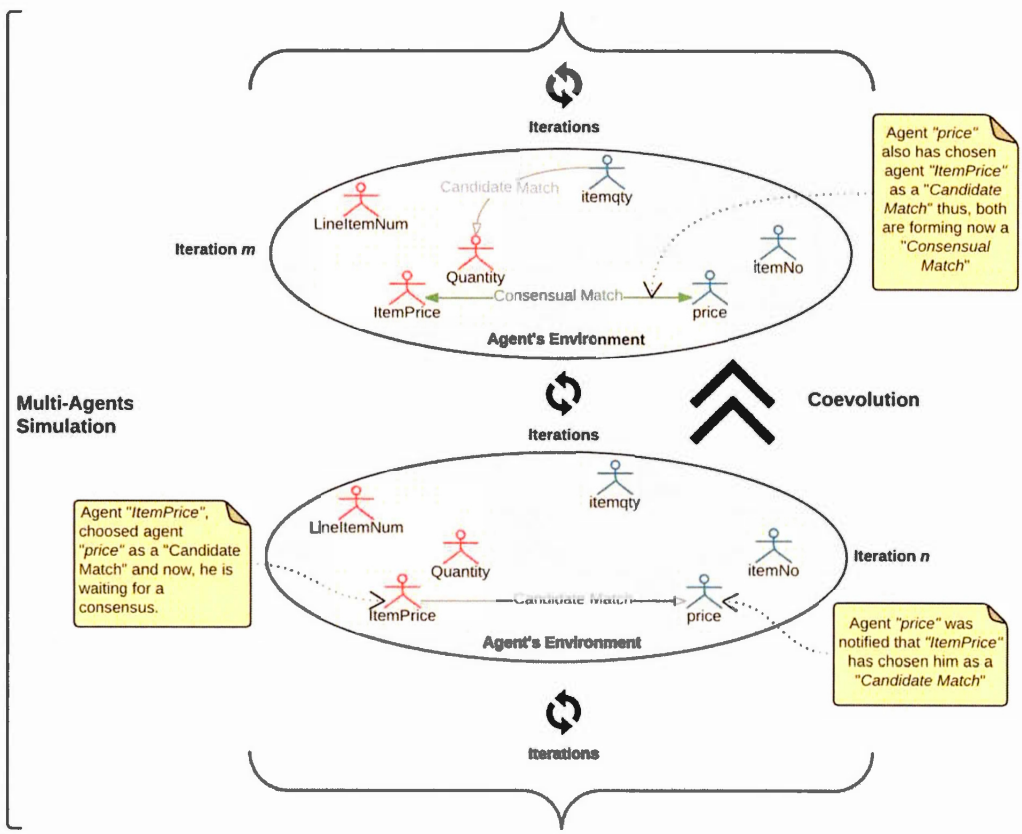


Figure 2.10 Appariement consensuel

2.4.2.4 Auto-organisation et émergence

Le mécanisme d'auto-organisation, dans le contexte des systèmes complexes, est le mécanisme par lequel l'émergence d'une solution globale (niveau macroscopique) se produit à partir des interactions au niveau local (niveau microscopique) des agents et ce sans contrôle central.

Touchant à notre modèle, ce mécanisme d'auto-organisation est le mécanisme qui opère au cœur de notre modèle, pour l'émergence d'une solution à l'appariement de schémas. Il peut être illustré par la figure 2.11. Cette dernière montre comment les propriétés de l'auto-organisation et de l'émergence se manifestent dans notre approche pour la résolution d'un scénario d'appariement. Cela revient à dire que dans notre modèle, la résolution du problème de l'appariement passe par l'émergence d'une auto-organisation globale à partir des interactions locales, gouvernées à partir de règles simples des agents.

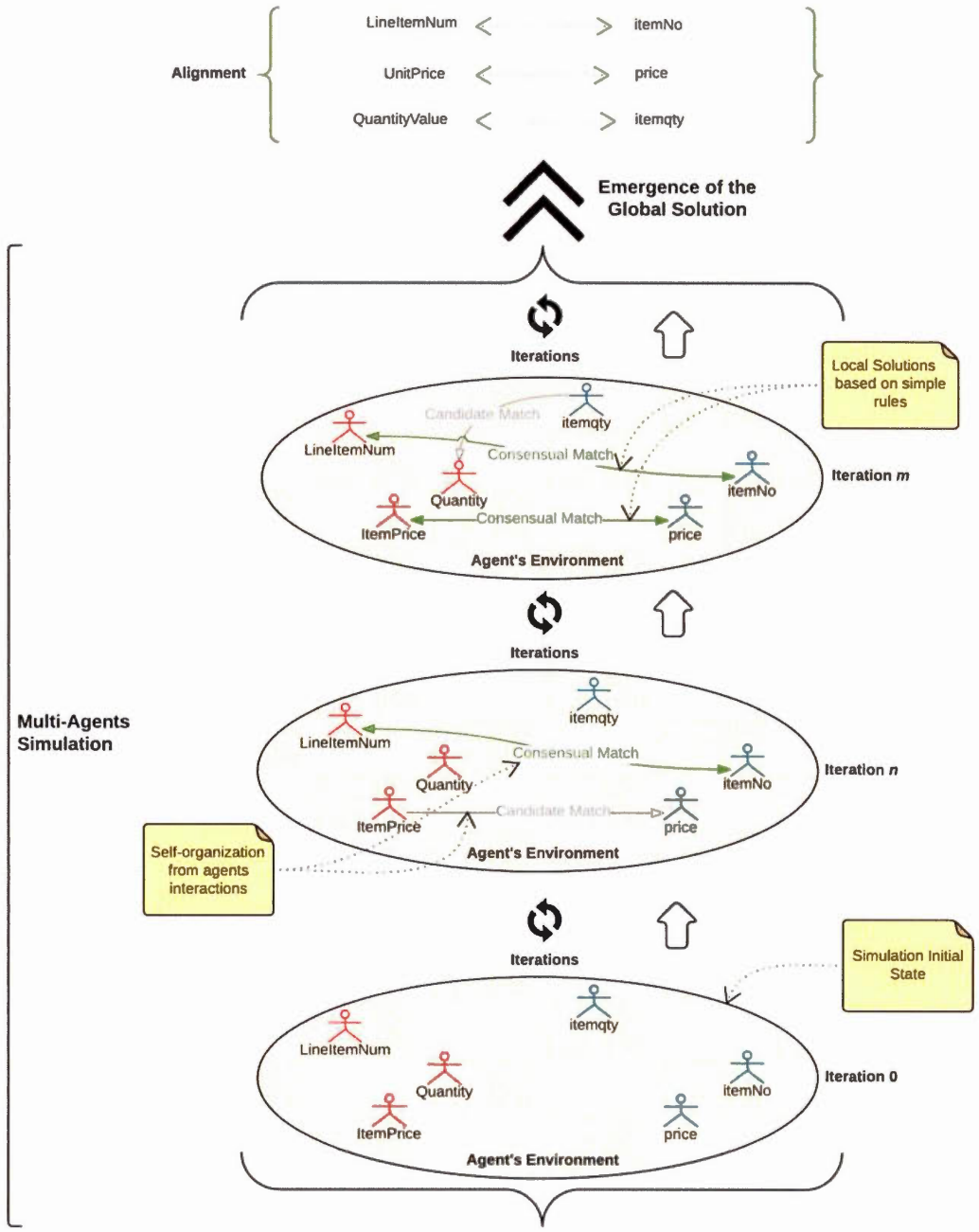


Figure 2.11 Résolution du problème d'appariement par émergence

En effet, pour nous, la notion d'émergence traduit l'apparition d'une nouvelle structure, par l'entremise du mécanisme de l'auto-organisation, à savoir un réseau reliant les agents du premier groupe (représentant le schéma source) aux agents du deuxième groupe (représentant le schéma cible). Cette émergence d'une structure auto-organisée est aussi le produit d'une recherche stochastique dynamique de chaque agent dans son espace de recherche. Cette recherche stochastique fait que tout le système évolue vers un état d'optimum global pour le problème traité (résolution de problèmes par un système composé d'éléments en interaction). Cette notion d'émergence traduit, d'autre part, la non prédictibilité de la formation de cette nouvelle structure à partir des conditions initiales.

2.4.2.5 Simulations multiples et analyse statistique

Dans l'optique d'aller encore plus loin dans la quantification et la réduction de l'incertitude, on s'est inspiré de la méthode *Simulation Monte-Carlo* en ce qui a trait à l'idée de l'application de l'analyse statistique sur des simulations stochastiques sans pour autant l'adopter entièrement. En effet, contrairement à la méthode *Simulation Monte-Carlo*, dans le contexte de notre modèle on ne cherche pas nécessairement à trouver, à partir des résultats des simulations, une distribution sur un paramètre donné ni à évaluer notre modèle. Dans la littérature, on parle aussi de simulation stochastique : « *La simulation stochastique qui est une application de type Monte Carlo, consiste à évaluer le modèle en le faisant vivre ou fonctionner d'une façon réaliste et comportant un ou plusieurs aspects aléatoires* » (Laurencelle, 2001).

Notre modèle, avec tous les concepts fondamentaux qu'il intègre, a été pensé depuis le début avec l'idée qu'une seule simulation serait amplement suffisante pour participer à une réduction de l'incertitude. Nous avons constaté cependant, lors de nos premières itérations, deux observations qui nous ont motivés à voir ce concept d'analyse statistique sur des simulations multiples (*méta-simulation*), ajouté à notre modèle.

La première observation, concerne la formation, de temps à autre, d'appariements consensuels incorrects et ce malgré les multiples, garde-fous enchâssés dans notre modèle, pour empêcher ces erreurs. En effet, dans notre modèle, lors de chaque itération de la simulation, chaque agent exécute toutes les étapes d'un processus d'appariement générique. En d'autres termes, l'agent durant sa durée de vie (durant la durée de la simulation) peut exécuter des dizaines, si ce n'est pas des centaines de fois, un processus d'appariement complet dans le but trouver une correspondance stable entre lui et un agent du groupe opposé (appariement consensuel). On a vu que grâce à l'appariement consensuel, le mauvais choix d'un agent lors d'une simulation, dans la plupart des cas, ne peut pas constituer un danger pour la formation d'une correspondance incorrecte. Signalons en passant que l'on sait pertinemment qu'il n'existe pas de modèle d'appariement automatique pouvant éviter totalement des erreurs d'appariement. Et donc, il est évident, vu l'incertitude inhérente à l'appariement automatique, que l'on ne peut pas avoir la prétention de dire que notre modèle pourrait être capable de vaincre totalement cette incertitude.

La seconde observation concerne la variabilité de la durée de simulation (i.e. nombre d'itérations). En effet, quelque fois, la simulation peut atteindre le nombre maximal des itérations sans que toutes les correspondances possibles aient été trouvées. Il opportun de signaler que pour des raisons de performance, on limite la durée maximale de la simulation à l'aide d'un paramètre (e.g. 3000 itérations).

Une analyse statistique nous permet de consolider le résultat des différentes simulations (constituant la méta-simulation), et ainsi de quantifier l'incertitude concernant ces résultats, en calculant la fréquence de l'apparition d'une correspondance au niveau de l'ensemble des simulations. L'exécution de simulations multiples (méta-simulation) peut nous aider à aller au-delà de la quantification de l'incertitude concernant les correspondances pour nous permettre l'étude, vu le grand volume et la grande richesse des données générées par les différentes simulations, de la performance des différentes

composantes du système (e.g. la performance des différents appariеurs). Toutefois, l'exécution de simulations multiples vient avec un prix à payer au niveau de la performance (peut être assez coûteuse au niveau temps de réponse). Ce prix peut être, cependant, réduit avec le recours au parallélisme (exécution parallèle des simulations sur plusieurs machines ou sur des microprocesseur multi-cœur).

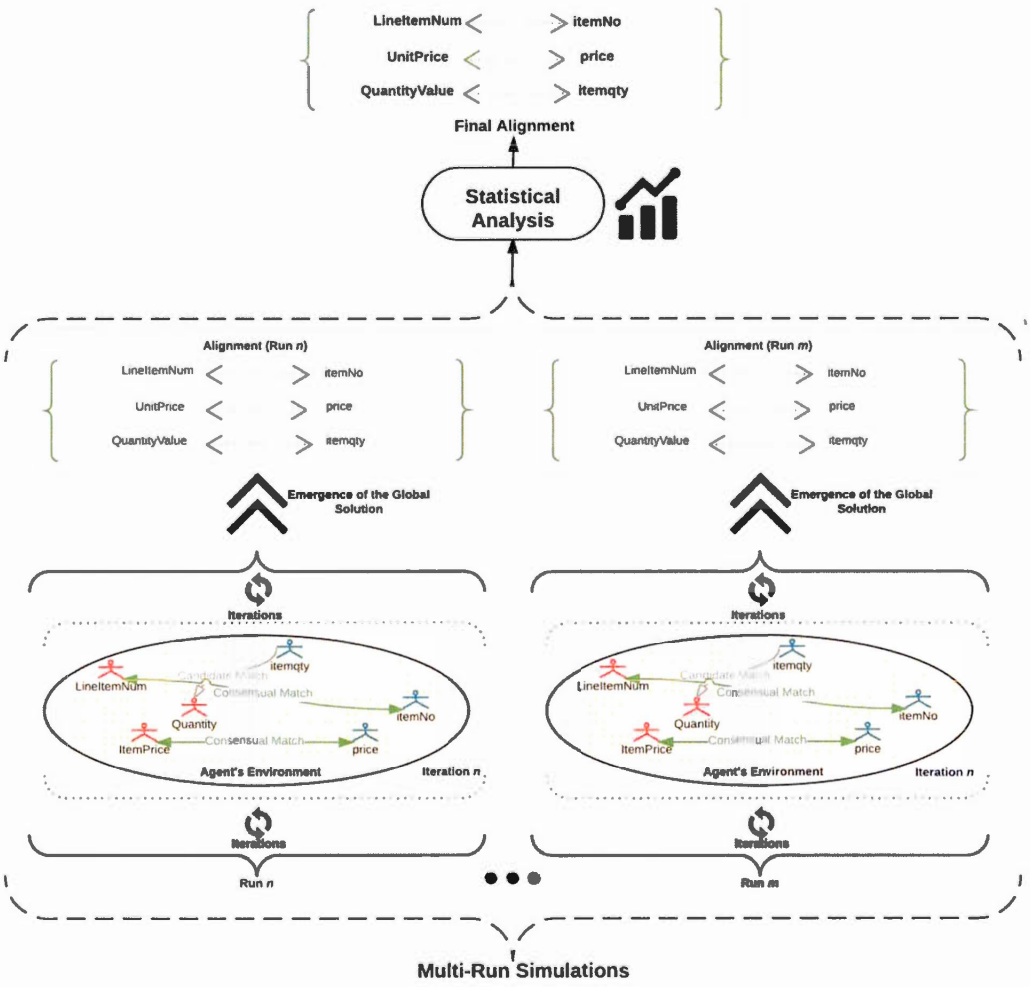


Figure 2.12 Analyse statistique sur le résultat de plusieurs simulations

De toutes ces considérations, il ressort comme le montre la figure 2.12, que l'idée de l'introduction de l'analyse statistique sur le résultat de plusieurs simulations s'imposait d'elle-même. Il est indéniable que ce concept s'intègre parfaitement dans la logique de notre modèle de simulation multi-agents pour l'appariement de schémas.

2.4.3 Transcription du modèle

La transcription du modèle consiste en une transformation du modèle conceptuel en un programme informatique s'exécutant sur une plateforme de simulation. Nous proposons dans cette thèse un prototype (i.e. *Reflex-SMAS*) basé sur des agents de type réactif. De plus nous présentons une proposition d'évolution, sur le plan conceptuel, concernant le comportement de l'agent (i.e. *agent élément de schéma*) pouvant donner lieu à une seconde version du prototype (i.e. *Rational-SMAS*) basée sur les agents de type cognitif/rationnel.

La figure ci-dessous montre l'évolution de notre approche/outil tout au long de la recherche.

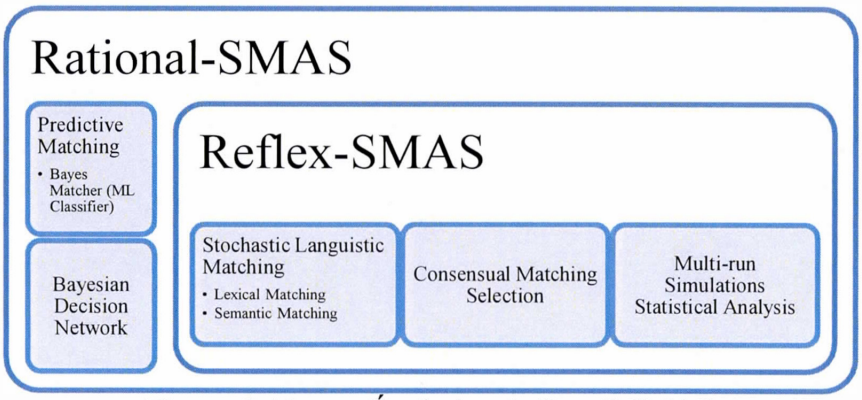


Figure 2.13 Évolution de l'outil SMAS

2.5 Conclusion

Dans la première partie de ce chapitre, nous avons présenté de façon détaillée le cadre théorique dans lequel on se place pour aborder la problématique de l'appariement automatique de schémas, ainsi que notre cheminement méthodologique.

Dans la seconde partie, nous avons proposé une formalisation du modèle conceptuel de notre approche de simulation multi-agents pour l'appariement automatique de schémas.

Le prochain chapitre se consacre à la description détaillée du modèle opérationnel de la version réflexive de notre modèle conceptuel. On y découvre la transcription des concepts fondamentaux de notre modèle conceptuel en un programme de simulation informatique.

CHAPITRE III

SIMULATION D'AGENTS RÉFLEXIFS (REFLEX-SMAS)

3.1 Introduction

Dans le présent chapitre sont exposées les étapes de la transcription de notre modèle conceptuel (décrit dans le précédent chapitre), *Modélisation et simulation à base d'agents pour l'appariement des schémas (SMAS)*, en un prototype de *Simulation d'agents réflexifs pour l'appariement des schémas (Reflex-SMAS).*

Dans un premier temps, nous exposons en détail l'implantation du prototype *Reflex-SMAS,* tout en accordant une attention particulière à l'architecture interne de l'agent réflexif (l'appariement du point de vue de l'agent). Dans un second temps, nous présentons et discutons le résultat des expérimentations, basées sur l'exemple de référence de notre prototype.

En dernier lieu, nous présentons à l'aide d'exemples, différentes façons de tirer avantage de la grande richesse des informations contenues dans les extrants de la simulation (i.e. données générées par les agents durant la simulation). À cet effet, nous introduisons une représentation ainsi qu'une analyse visuelle des interactions des éléments du système d'appariement durant une simulation permettant notamment une analyse de la performance des différents appariеurs pour un scénario d'appariement donné. Aussi, nous explicitons la conception d'un appariеur prédictif, basé sur l'apprentissage machine (*classificateur bayésien*), exploitant les données collectées et

générées lors de différentes simulations et permettant de prédire le meilleur appariement.

3.2 Modèle opérationnel

L'objectif de cette section est de décrire le modèle opérationnel (computationnel, exécutable) traduisant notre modèle conceptuel vers un programme informatique de *simulation multi-agents* pour l'appariement de schémas (i.e. prototype).

Avant d'aller plus loin, il serait indiqué de resituer la notion de modèle opérationnel dans le contexte de *la modélisation et simulation à base d'agents*. Reprenons à cet effet, la définition du modèle opérationnel avancée par (Treuil *et al.*, 2008) : « *Modèle dynamique spécifié dans un langage qui respecte le méta-modèle associé à un simulateur, et qui peut être directement interprété ou exécuté par ce simulateur* ».

Le modèle opérationnel, ou encore le modèle exécutable, doit donc décrire les spécifications nécessaires à son implémentation en un programme informatique pouvant être exécuté par la plateforme de simulation. C'est ainsi que, concernant notre modèle opérationnel, nous allons nous concentrer essentiellement sur les spécifications des composantes principales de ce modèle, notamment l'agent réactif (i.e. *SEAgentReflex*).

Ces spécifications sont ensuite traduites en un programme informatique, de *simulation multi-agents*, en se basant sur le paradigme de la *programmation orientée-agents*[68]. Ce dernier pourrait être considéré comme une spécialisation du paradigme de la *programmation orientée-objets*[69] (Shoham, 1993). En effet, on pourrait dire que la *programmation orientée-agents* et celle de *l'orientée-objets* s'apparentent dans la mesure où ces deux paradigmes se basent sur un ensemble de modules, qui

[68] De l'anglais « *agent-oriented programming (AOP)* »

[69] De l'anglais « *object-oriented programming (OOP)* »

communiquent avec des messages, et qui sont capables de traiter ces mêmes messages chacun à sa façon (Shoham, 1993). Toutefois, les objets, contrairement aux agents, n'exhibent pas la capacité de l'autonomie. Avec cette dernière, l'agent peut être considéré comme un *objet,* ayant un comportement autonome, capable de raisonner sur son état (i.e. croyances sur le monde), dans l'objectif de décider des actions à prendre pour arriver à un but donné.

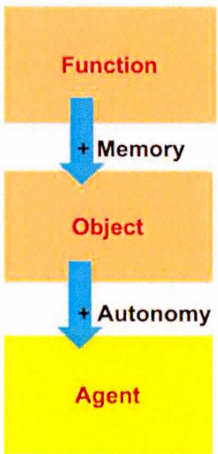


Figure 3.1 C'est quoi un « Agent » **(Macal et North, 2006)**

La figure 3.1, nous offre une perspective intéressante sur l'évolution de la *programmation procédurale* vers *la programmation orientée-agents*, en passant par *la programmation orientée-objets*, nous permettant ainsi d'établir des parallèles entre ces différents paradigmes et d'identifier rapidement les spécificités principales pouvant les caractériser.

Dans la même veine, (Siegfried, 2014) souligne, dans le contexte des simulations informatiques, l'évolution de la *simulation orientée-objets,* qui consiste fondamentalement en un ensemble d'objets interagissant dans le temps, vers la *simulation orientée-agents*. Selon (Siegfried, 2014), la *simulation orientée-objets*, dont les racines remontent aux années 1960, serait en quelques sorte l'ancêtre de la

simulation orientée-agents. Or, contrairement à la *simulation orientée-agents*, dans une *simulation orientée-objets*, les *objets* ne disposant pas de la capacité d'autonomie, se retrouvent alors, contraints à des comportements purement réactifs.

Concernant la *simulation orientée-agents*, soulignant au passage, une confusion qu'il conviendrait de clarifier. Dans la littérature, on réfère, quelques fois, au concept de *simulation multi-agents*[70] avec le terme *systèmes multi-agents*[71]. Or, malgré le fait qu'il est tout à fait correct d'utiliser le terme *systèmes multi-agents* pour désigner une *simulation multi-agents,* il subsiste tout de même une nuance. Cette nuance étant qu'une *simulation multi-agents* est forcément un *système multi-agents,* mais que le contraire n'est pas toujours vrai. En effet, une *simulation multi-agents* est bel et bien un *système multi-agents*, dont la vie se déroule dans un monde artificiel (notamment dans un simulateur), ce qui n'est pas forcément le cas pour tous les *système multi-agents* qui peuvent pour certains, représenter ou même être eux même, des entités du monde réel (e.g. des robots).

L'articulation de cette section, décrivant le modèle opérationnel, va se situer autour des trois grandes composantes qui ressortent dans la définition de la *modélisation et simulation à base d'agents* avancée par (Macal et North, 2006, p. 24 What Is Agent-Based Modeling & Simulation?), à savoir les agents, les relations et la plateforme de simulation : « *What Is Agent-Based Modeling & Simulation ? An agent-based model consists of: (i) a set of agents (part of the user-defined model), (ii) a set of agent relationships (part of the user-defined model), and (iii) a framework for simulating agent behaviors and interactions (provided by an ABMS toolkit or other implementation).* ».

---

[70] De l'anglais « *multi-agent simulation* »

[71] De l'anglais « *multi-agent system (MAS)* »

3.2.1 Agent élément de schéma (*SE-Agent*)

Comme on l'a déjà explicité auparavant, la modélisation des simulations multi-agents se base en général sur la modélisation des agents dont les interactions « *locales* », tout au long de la simulation, conduisent à faire émerger un ordre « *global* ». En somme, on peut dire, qu'en règle générale, la modélisation multi-agents pourrait être définie, jusqu'à un certain point, en termes de comportements individuels des agents participant à la simulation. Notre modèle, de la même manière, ne fait pas exception à cette règle dans le sens ou la composante centrale de notre modèle de ***simulation à base d'agents*** est bien évidement ***l'agent élément de schéma*** (i.e. *SEAgentReflex* qui se trouve être une spécialisation de l'agent *SEAgent*). Le comportement de l'agent *SEAgentReflex* peut être décrit, en excluant l'étape d'initialisation de la simulation, comme un cycle à trois phases : d'abord l'agent commence ce cycle par la phase de (i) perception de son environnement, ensuite et en se basant sur l'ensemble des percepts, l'agent entame la phase de (ii) la prise de décision concernant les actions à prendre, et enfin et après avoir sélectionné l'action à prendre, l'agent complète ce cycle par la phase de (iii) l'exécution des actions pendant laquelle l'agent va poser des gestes concrets modifiant ainsi son état ou l'état de son environnement.

3.2.1.1 Perception

Dans le contexte de notre modèle opérationnel, l'agent *SEAgentReflex* durant la phase de perception, perçoit son environnement (non pas à l'aide de *capteurs*[72] proprement dit) en interrogeant son environnement, en effectuant des calculs de similarité (que l'on peut considérer comme un acte de reconnaissance) ou encore en capturant certains évènements. Le résultat de cette phase va être un ensemble de percepts, permettant à l'agent, non seulement d'identifier les agents de l'autre groupe, disponibles pour l'appariement, mais encore le degré de similarité textuelle (i.e. similarité lexicale), qui

[72] De l'anglais « *sensors* »

existe au niveau de l'attribut *name* ainsi qu'au niveau de l'attribut *comment*, entre cet agent et les autres agents du groupe opposé.

Pour chaque agent du groupe opposé, l'agent commence le calcul de similarité lexicale (pour l'attribut *name* et l'attribut *comment*), en tirant une mesure de similarité au hasard (générée aléatoirement par l'environnement à partir d'un bassin de mesures). Cette mesure calcule et retourne le score de similarité qui est ensuite comparé à un seuil aléatoire (une valeur aléatoire à l'intérieur d'un intervalle, retournée par l'environnement). Dans le cas où le score dépasse le seuil, et éventuellement supérieur au meilleur score obtenu lors des itérations antérieures, alors le score obtenu est considéré comme meilleur score (pour l'attribut *name* ou pour l'attribut *comment,* selon l'étape de perception).

$$\textit{Un score de similarité est considéré comme le meilleur score, si le score obtenu est:} \begin{cases} \textit{le plus haut score parmi les score obtenus pour tous les agents du groupe oppose} \\ \textit{supérieur à un seuil généré aléatoirement} \\ \textit{supérieur au meilleur score deja obtenu lors des iterations precedentes} \end{cases}$$

(3.1) Règle de la phase de perception : calcul de similarité pour le nom ou le commentaire

Outre les actions, de l'interrogation de l'environnement ou du calcul de similarités, permettant à l'agent de percevoir son environnement, la capture d'évènements, provenant de l'environnement, est une autre action de perception de la part de l'agent. L'agent procède au filtrage des évènements déclenchés autour de lui dans le but de cibler et de capturer un évènement bien précis, en l'occurrence l'évènement qui se déclenche lorsqu'il est choisi par un autre agent comme un appariement candidat.

3.2.1.2 Décision

Durant cette phase, l'agent à partir des résultats de la phase de perception (i.e. l'ensemble des percepts et le cas échéant la croyance actuelle de l'agent concernant le meilleur appariement) raisonne, délibère et décide sur l'action à sélectionner. Les grandes décisions, impliquant le choix d'actions, qui sont à prendre par l'agent plusieurs fois (plusieurs itérations de la même simulation) durant sa durée de vie, sont les suivantes : (i) la décision concernant la convergence des similarités *name* et *comment* et la sélection d'un appariement candidat, (ii) la décision concernant la réinitialisation des croyances concernant l'appariement candidat, et enfin, (iii) la décision concernant l'appariement consensuel.

La première décision est prise lorsque l'agent constate, lors du même cycle *perception-décision-action* (i.e. même itération de la simulation), une convergence des similarités *name* et *comment* vers le même agent du groupe opposé. En d'autres termes, pour que l'agent décide de sélectionner un agent du groupe opposé comme un appariement candidat, il faut d'abord qu'il réalise qu'une convergence existe, pointant vers le même agent du groupe opposé, entre l'agent sélectionné comme ayant la meilleure similarité au niveau de l'attribut *name* (i.e. le nom de l'élément de schémas) et celui sélectionné comme ayant la meilleure similarité au niveau de l'attribut *comment* (i.e. description de l'élément de schéma). Ensuite, et dans le cas où la convergence est avérée, l'agent demande une fonction d'agrégation (i.e. Max, Average, Weighted) à l'environnement (la fonction est tirée au hasard) à la suite de quoi, l'agent procède au calcul d'agrégation des deux scores (i.e. pour l'attribut *name* et l'attribut *comment*). Enfin, l'agent du groupe opposé est désigné comme appariement candidat, si le nouveau score agrégé est supérieur à un seuil aléatoire et éventuellement supérieur au meilleur score agrégé déjà obtenu lors des itérations antérieures.

$$\begin{cases} \textit{un agent du groupe opposé est considéré comme appariement candidat, si :} \\ \textit{convergence de la similarité, pour le nom et le commentaire, vers le même agent} \\ \textit{supériorité du score aggrégé par rapport au score du precedent appariement candidat} \end{cases}$$

(3.2) Règle de la phase de decision : convergence vers appariement candidat

La deuxième décision doit être prise lorsque l'agent réalise, au bout d'un certain temps (i.e. un nombre d'itérations basé sur un nombre aléatoire dans un intervalle), que l'attente d'un consensus est vaine (délai d'attente d'un consensus écoulé) et qu'il serait préférable de revenir en mode exploration. Avec cette décision, l'agent fait alors le choix de réinitialiser ces croyances concernant l'appariement candidat.

$$\begin{cases} \textit{Un agent doit effacer ces croyances concernant un appariement candidat, si :} \\ \textit{il a un appariement candidat} \\ \textit{depuis un nombre d'itérations égale à nombre aléatoire dans un interval } (e.g. 150 \textit{ et } 250) \end{cases}$$

(3.3) Règle de la phase de decision : délai écoulé

La dernière décision vient essentiellement en réponse à l'évènement qui vient notifier à l'agent qu'il a été choisi comme appariement candidat. L'agent doit alors vérifier si les conditions sont réunies pour former un appariement consensuel. En effet, la formation d'un appariement consensuel est conditionnée par une désignation mutuelle des deux agents comme appariement candidat.

$$\begin{cases} \textit{Un agent trouve le consensus avec un agent du groupe opposé, si :} \\ \textit{les deux agents ont un appariement candidat} \\ \textit{chaque agent designe l'autre agent comme appariement candidat} \end{cases}$$

(3.4) Règle de la phase de decision : consensus trouvé

On peut se rendre compte, que la phase de décision, de par la nature même de notre agent réflexif (i.e. agent réactif), ne comporte pas des tâches de raisonnement, de planification ou d'apprentissage. Cependant, on peut dire que lors de cette phase les agents peuvent faire le choix d'actions qui peuvent, à mesure que la simulation avance,

exhiber la capacité d'adaptation (en quelques sortes d'apprentissage). Cette dernière prend la forme de boucles de rétroactions où les agents peuvent revoir et revenir sur leurs décisions-actions défaillantes, prises lors des itérations antérieures (e.g. réinitialiser les croyances concernant l'appariement candidat après le découlement du temps d'attente d'un appariement consensuel). Cela permet de générer, pour l'agent, de nouvelles trajectoires dans l'espace des trajectoires possibles (e.g. utilisation de nouvelles mesures de similarité, de nouvelles fonctions d'agrégation). Plus que de l'adaptation, il s'agit là de la manifestation d'une autre caractéristique fondamentale d'un *système complexe adaptatif*, à savoir la coévolution *agent-agent* (dans la mesure où la décision d'un agent est tributaire de la décision d'un autre agent) et *agent-environnement* (dans la mesure où l'environnement impose des contraintes, comme par exemple le choix aléatoire des mesures de similarité, à l'agent qui à son tour modifie l'environnement en prenant des décisions et ensuite en les appliquant en effectuant des actions).

3.2.1.3 Action

Pendant la phase de l'action, l'agent exécute les actions sélectionnées lors de la précédente phase, en l'occurrence celle de la décision. L'itération courante de la simulation prend sa fin avec cette phase.

Durant cette phase, notre agent accomplit les actions sélectionnées lors de la phase de décision à savoir sélectionner un appariement candidat, effacer ses croyances concernant un appariement candidat, ou encore former un appariement consensuel. Pour effectuer ces actions, l'agent, à l'aide de ses effecteurs[73], change son état interne et passe la demande à l'environnement pour modifier le réseau des appariements, basé sur la topologie sous-jacente de l'environnement.

---

[73] De l'anglais « *effectors* »

En premier lieu, lorsque l'agent prend la décision concernant le choix d'un appariement candidat, il pose les gestes suivants : (i) il met à jour l'objet contenant les informations relatives à l'appariement candidat (i.e. la classe *CandidateMatchingAttribute*), ensuite (ii) il fait transiter son état interne de l'état *UNDEFINED_MATCH_STATE* à l'état *CANDIDATE_MATCH_STATE,* et enfin (iii) il procède à la création d'un arc dirigé, dans le réseau des appariements, vers l'agent qu'il a choisi comme appariement candidat. Le changement d'état provoque le déclenchement d'un signal, propagé par l'environnement, indiquant, au reste du monde (incluant l'agent concerné par le choix), que cet agent est prêt pour un appariement consensuel.

En second lieu, lorsque l'agent prend sa décision concernant l'écoulement du délai d'attente d'un consensus, il procède aux actions suivantes : l'agent (i) commence par effacer ses croyances concernant l'appariement candidat, et ce en réinitialisant l'objet contenant les informations relatives à l'appariement candidat (i.e. la classe *CandidateMatchingAttribute*), ensuite il (ii) passe au changeant de son état interne de l'état *CANDIDATE_MATCH_STATE à* l'état *UNDEFINED_MATCH_STATE,* et enfin, il (iii) procède à la suppression de l'arc dirigé, dans le réseau des appariements, pointant vers l'agent qu'il avait choisi comme appariement candidat .

En dernier lieu, lorsque l'agent prend sa décision concernant la formation d'un appariement consensuel, il pose les gestes suivants : (i) met à jour l'objet contenant les informations concernant l'appariement consensuel (i.e. la classe *ConsensualMatchingAttribute*), ensuite (ii) fait transiter son état interne de l'état *CANDIDATE_MATCH_STATE* à l'état *CONSENSUAL_MATCH_STATE.* Une fois ce nouvel état atteint, l'agent rentre alors dans un état irréversible (i.e. *STABLE_MATCH_STATE=true*) qui sonne la fin du cycle *perception-décision-action* de l'agent (la fin de la phase d'exploration de son environnement à la recherche d'un appariement), tout en gardant l'agent en vie jusqu'à la fin de la simulation, et enfin (iii)

procède à la création d'un arc dirigé, dans le réseau des appariements, vers l'agent avec lequel il forme un appariement consensuel.

3.2.2 Relations

Après *les agents*, vient l'autre principale composante, pour la *modélisation et la simulation à base d'agents*, à savoir *les relations et les interactions* entre les agents. Ces relations sont généralement régies par des règles de comportement, en général simples. Outre les règles, inscrites dans la logique interne des agents, les interactions s'exercent aussi sous l'influence de l'environnement (e.g. topologie, contraintes), dans lequel tiennent lieu ces interactions.

Concernant notre modèle, le but ultime de ces interactions est la découverte des relations de correspondances entre les éléments de schémas. Dans cette perspective, l'agent participe aux interactions avec les autres agents, à la fois d'une façon implicite et explicite. D'un côté, l'interaction implicite tient lieu lorsque l'agent prend part à une interaction sans qu'il en soit explicitement informé (e.g. lorsqu'il fait partie du traitement de la phase de perception d'un autre agent). De l'autre côté, l'interaction explicite tient lieu lorsque l'agent provoque lui-même, d'une manière unilatérale ou consensuelle, une interaction (e.g. la création d'une relation « *asymétrique* » d'appariement candidat, ou encore la création d'une relation « *symétrique* » d'appariement consensuel), par l'entremise d'une communication indirecte (i.e. *stigmergie*).

Il est à noter que malgré l'interaction constante avec son environnement, notre agent en a une connaissance limitée, dans la masure où il ne perçoit pas les choix des autres agents. Une des conséquences de cette connaissance limitée réside dans le fait que l'agent peut exhiber un comportement irrationnel dans la mesure où il peut sembler s'entêter à sélectionner un agent comme un appariement candidat alors que ce dernier a déjà formé un appariement consensuel avec un autre agent. Il convient de préciser

que nous avons choisi sciemment de ne pas éliminer les agents (*élagage de schémas*[74]) ayant pu former un appariement consensuel, des calculs effectués lors de la phase de perception. La raison derrière la décision de ne pas faire d'élagage de schémas, afin de réduire l'espace de recherche et ainsi éviter des calculs inutiles, tient en notre volonté d'éviter de pousser l'agent vers un appariement par défaut. Autrement dit, on préfère voir se manifester, lors d'une simulation, un comportement « irrationnel » (i.e. d'entêtement à choisir, comme appariement candidat, un agent non disponible), nous permettant peut-être de tirer des enseignements de cette situation, plutôt que de voir ce comportement disparaitre avec l'élagage.

Signalons en passant que pour notre modèle conceptuel et opérationnel, nous avons choisi de ne pas modéliser des interactions entre l'agent et les autres agents du même groupe (i.e. schéma source ou schéma cible). Cependant, on pourrait très bien envisager, dans des évolutions futures de notre modèle, ce genre d'interactions et ce dans l'optique de créer des relations de *parenté*[75] qui pourraient servir pour la résolution de la problématique des appariements complexes[76] (i.e. appariements de cardinalité $n:m$).

Enfin, il est important de noter qu'il est préférable de garder les règles d'interaction les plus simples possibles. Ceci a l'avantage de contribuer à la facilité de compréhension et d'analyse du modèle.

---

[74] De l'anglais « *schema pruning* »

[75] De l'anglais « *relatedness* »

[76] De l'anglais « *complex matching* »

3.2.3 Plateformes

En règle générale, pour faire exécuter des simulations on doit avoir recours à des simulateurs. Un simulateur est un programme (ou plateforme) informatique capable d'interpréter des modèles dynamiques, et utilisé(e) pour produire les perturbations désirées sur ces modèles (Treuil *et al.*, 2008).

Avant d'aller plus loin, il y aurait intérêt à présenter les principales plateformes de simulation. Comme l'illustre la figure 3.2, dans le domaine de la modélisation et de la simulation multi-agents, une multitude de plateformes et d'outils existent pour le développement de simulations multi-agents. Ces outils peuvent se catégoriser selon leur facilité et leur puissance de développement. Ainsi, on peut voir que l'outil *NetLogo* (Sklar, 2007 ; Tisue et Wilensky, 2004), est considéré comme un outil offrant à la fois une facilité et une puissance modérée de développement. Par contre, on peut voir que l'outil *Repast Simphony* (North *et al.*, 2007), basé sur le langage *Java* et sur l'environnement de développement intégré (IDE) *Eclipse*, offre plus de puissance et de flexibilité pour le développement, moyennant un coût au niveau de la simplicité d'utilisation.

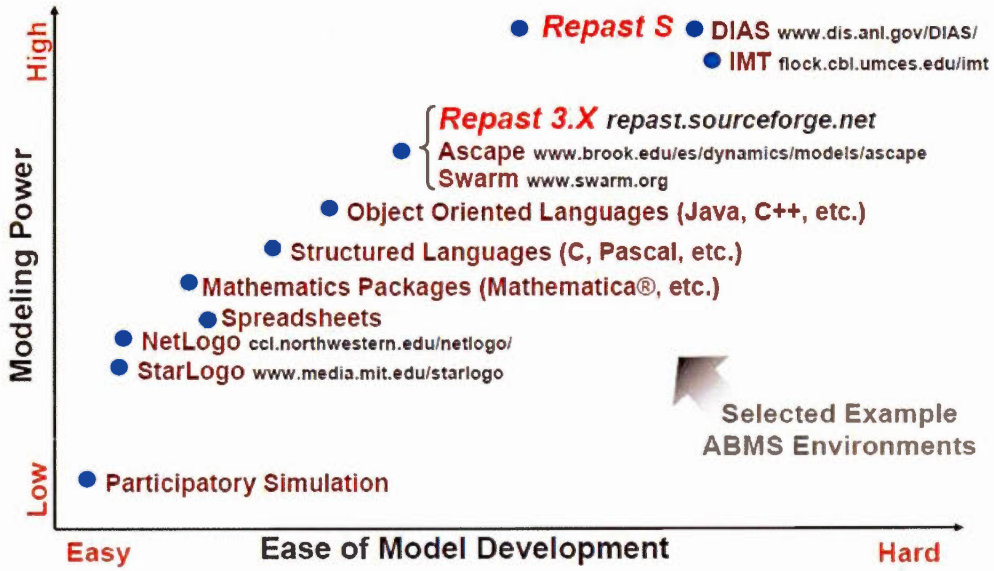


Figure 3.2 Plateformes de simulation multi-agents **(Macal et North, 2006)**

Au début de notre cheminement de recherche, en raison de sa simplicité, le choix a été porté sur l'outil *NetLogo* pour le développement d'une sorte de maquette (prototype jetable), de notre modèle de simulation multi-agents pour l'appariement de schémas. Cette maquette avait pour objectif, dans un premier temps, la validation des grandes lignes de notre approche de façon à avoir un retour rapide (confirmation préliminaire de nos intuitions) sur notre hypothèse centrale, à savoir modéliser l'appariement automatique de schéma sous la forme de simulations multi-agents.

Une fois le résultat positif de cette première étape confirmé, le choix d'une plateforme plus avancée, pour l'implémentation de notre prototype, a été porté sur l'outil *Repast Simphony* (North *et al.*, 2007).

3.3 Prototype et expérimentations

Dans cette section, nous présentons le prototype implémentant notre modèle opérationnel, à la suite de quoi nous présentons et discutons les expérimentations préliminaires faites ce sur ce prototype.

Bâtir un prototype, était pour nous un choix inévitable dans la mesure où une validation empirique non seulement de notre hypothèse centrale mais aussi de notre modèle conceptuel était nécessaire. Dans ce contexte, le terme *simulation* pourrait devenir alors, une *polysémie* dont il conviendrait de démêler les différents sens. En effet, on parle de *simulation* dans le contexte de notre prototype, d'un côté, dans le sens d'un outil expérimental, une simulation informatique, ayant comme objectif le test, la vérification et le cas échéant l'accréditation de notre hypothèse centrale (i.e. appariement de schémas comme système complexe adaptatif). De l'autre côté, on parle de *simulation* dans le sens du produit de notre prototype (i.e. *simulation multi-agents pour l'appariement de schémas*), ayant pour objectif, en simulant des interactions entre les éléments de schémas représentés comme des agents, de fournir la solution à un scénario d'appariement donnée. Quel que soit le sens que l'on entend par *simulation*, dans le contexte de notre prototype, le recours à la puissance des ordinateurs est inéluctable, comme le fait remarquer (Choi et Kang, 2013) : « *if our brain is not powerful enough to simulate a given complex system, we rely on computers to perform a computer simulation* ».

Les expérimentations faites dans le contexte des tests préliminaires sur notre prototype, ont suivi un processus itératif, visant principalement une vérification sommaire du modèle opérationnel ainsi qu'une une validation sommaire du modèle conceptuel. Une vérification et une validation exhaustive du modèle conceptuel et opérationnel seront présentées ultérieurement dans un chapitre à part (le chapitre 4 a été consacré à la validation de notre modèle et par conséquent la validation des hypothèses avancées dans cette thèse).

Pour nous, la vérification et la validation de notre modèle conceptuel ainsi que notre modèle opérationnel, passent par le développement d'un prototype. Ce dernier, devient, sans aucun doute, la pierre angulaire, de la vérification de la transcription du modèle conceptuel vers le modèle opérationnel, et de la validation de l'hypothèse centrale concernant l'utilisation de la simulation multi-agents pour l'appariement de schémas.

3.3.1 Prototype

3.3.1.1 Architecture haut-niveau

Notre prototype se base sur trois composantes principales, dont l'implémentation a été réalisée en privilégiant l'exploitation d'outils gratuits à code source ouvert[77].

En premier lieu, et concernant l'implémentation du cœur du prototype, à savoir notre modèle opérationnel de simulation multi-agents pour l'appariement de schémas *SMAS* (plus spécifiquement l'agent *SEAgentReflex*), le choix a été porté sur la plateforme *Repast Simphony* (la version 2.1) (North *et al.*, 2007). Outre la puissance et la flexibilité de cette plateforme, plusieurs autres raisons sont venues s'ajoutées pour justifier ce choix, par exemple, le fait que le langage *Java*[78] soit le langage principal de programmation pour cette plateforme, ou encore le fait que le code source de cette plateforme soit ouvert.

En deuxième lieu, et en ce qui a trait au calcul de similarité, notre choix s'est porté sur le cadriciel *DkPro* (version 2.1.0) (Bär *et al.*, 2013). Ce dernier, comme on l'a déjà mentionné, est un cadriciel qui implémente, avec le langage *Java*, une grande variété de mesures de similarité, à savoir des mesures textuelles, sémantiques, structurelles, stylistiques ou encore phonétiques dans un même et unique référentiel. Notons au

---

[77] De l'anglais « *open source* »

[78] *Repast Simphony* supporte aussi le langage *ReLogo* (pour la création rapide et facile des modèles de simulation) ainsi que le langage *C++* (pour la création des modèles de simulation necessitant une tres haute performance)

passage que ce cadriciel englobe des mesures provenant de deux autres librairies bien connues dans la littérature, en l'occurrence *SecondString* (W. W. Cohen *et al.*, 2003) et *SimMetrics* (Chapman, 2005).

En dernier lieu, et concernant la composante responsable de l'analyse statistique, la sélection du langage de programmation *R* (version R 3.1.0) (Ripley, 2001) a été un choix qui s'imposait de lui-même, vu l'intégration entre l'outil *Repast Simphony* et le langage *R* (North, 2010).

La figure 3.3 décrit l'architecture haut-niveau du prototype *SMAS*.

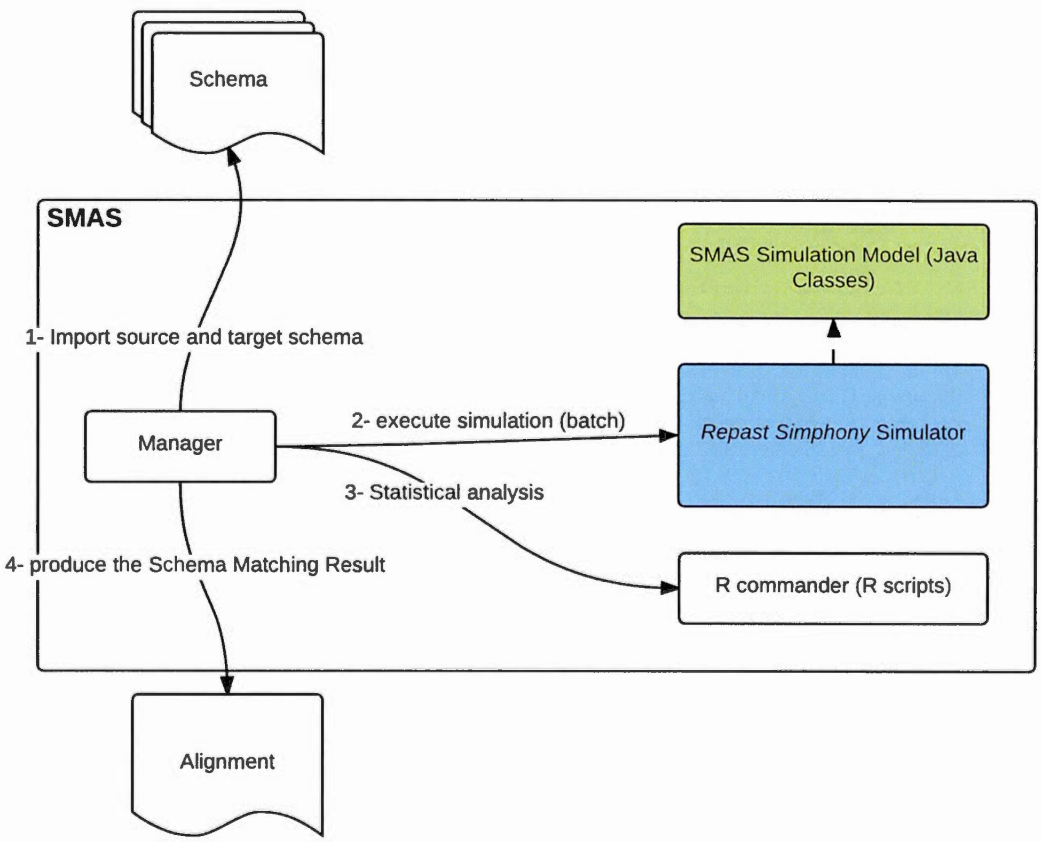


Figure 3.3 Architecture du prototype SMAS

3.3.1.2 Spécifications détaillées

Pour la conception de notre système multi-agents, nous avons opté pour le langage *UML (Unified Modeling Language)*. La prise en charge de la modélisation des systèmes multi-agents (Guedes et Vicari, 2009) avec le langage *UML* a été rendue possible avec l'*extension agents pour UML*[79] ou encore avec les nouvelles notations concernant les agents, de la version *UML 2.0* (Odell *et al.*, 2000 ; Bauer *et al.*, 2001 ; Huget, 2004 ; Guedes et Vicari, 2009). L'adoption de la notation de modélisation *UML*, pour la modélisation des *agents*, a pour avantage d'être totalement basée sur un langage de modélisation largement accepté dans l'ingénierie logicielle. Par le fait même, il permet de simplifier le passage de *l'ingénierie logicielle* [80] vers *l'ingénierie des systèmes multi-agents*[81] (Huget, 2004).

En outre, l'adoption du langage *UML*, comme notation de modélisation et de conception de notre système multi-agent a été principalement justifiée par la promesse d'une courbe d'apprentissage moins raide. En effet, la modélisation orientée agents, du point de vue des fondements, de l'approche ou encore de la notation, devient, avec le langage *UML*, une simple spécialisation de la modélisation orientée objets. C'est ainsi que nous avons été en mesure de tirer avantage des diagrammes offert par le langage *UML* pour modéliser aussi bien les aspects statiques que les aspects dynamiques de notre système multi-agents.

Ainsi, concernant l'aspect statique, le diagramme de classes a permis la modélisation de l'architecture interne de l'agent (i.e. ses attributs et ses comportements). La modélisation de l'aspect dynamique, quant à elle, a reposé d'un côté, sur le diagramme d'activités, pour la modélisation du comportement global de l'agent, et de l'autre côté,

---

[79] De l'anglais « agent UML (AUML) »

[80] De l'anglais « *software engineering* »

[81] De l'anglais « *multiagent system engineering* »

sur le diagramme d'états-transitions pour la modélisation de la transition des états internes de l'agent. Par ailleurs, nous n'avons pas eu recourt au diagramme de séquences pour la modélisation des interactions entre agents, d'une part parce que toutes les interactions entre agents se passent par l'entremise de l'environnement et d'autre part à cause de la faible valeur ajoutée (avec la présence du diagramme d'activités) pour l'amélioration de l'intelligibilité de notre modèle.

Concernant le diagramme de classes, la figure 3.4 représente les principales classes de notre prototype, notamment la classe *SEAgent* représentant la classe parent de l'agent réflexif (i.e. *SEAgentReflex*), ainsi que toutes les autres classes subséquentes.

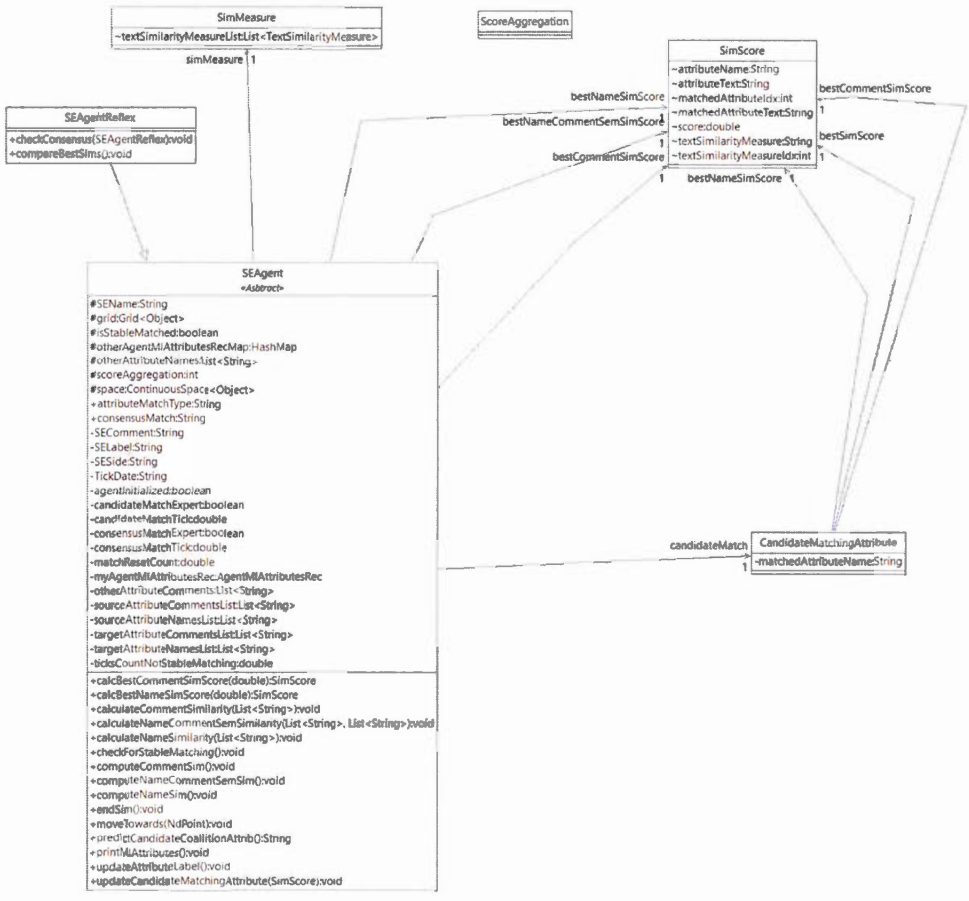


Figure 3.4 Diagramme de classes pour *Reflex-SMAS*

La classe *SEAgent*, est une classe abstraite représentant le concept *agent élément de schéma*. Cette classe implémente les propriétés, les règles ainsi que le comportement de l'agent, nonobstant le type de l'agent réflexif (i.e. *SEAgentReflex*) ou rationnel (i.e. *SEAgentRational*), permettant ainsi la réutilisation et le partage des caractéristiques communes entre les deux types. Il va sans dire, que la simulation ne référence pas directement la classe *SEAgent* (non-instanciable), mais référence plutôt les classes implémentant cette classe abstraite notamment la classe *SEAgentReflex* ou la classe *SEAgentRational.* Soit dit en passant, la seule différence notable entre cette première version du prototype (i.e. *Reflex-SMAS*) et la deuxième version du prototype (i.e. *Rational-SMAS)*, présentée en détail dans le prochain chapitre, est l'utilisation par la simulation de la classe *SEAgentRational*, implémentant les comportements d'un agent rationnel, au lieu de la classe *SEAgentReflex.*

La classe *SEAgentReflex* est donc une classe enfant implémentant les méthodes abstraites de la classe abstraite *SEAgent.* La classe *SEAgentReflex* représente une spécialisation implémentant les comportements d'un agent réflexif.

La classe *CandidateMatchingAttribute* est référencée par la classe *SEAgentReflex*, dans le cadre de l'action concernant la sélection d'un appariement candidat, afin de représenter et de contenir toutes les informations concernant cet appariement candidat. En effet, la classe *CandidateMatchingAttribute* permet de stocker, d'un côté, les informations concernant l'autre agent (i.e. l'index et le nom de l'autre agent) et de l'autre côté, grâce à la classe *SimScore*, les informations concernant les scores obtenus (le score obtenu pour le nom, celui obtenu pour le commentaire, ainsi que l'agrégation des deux).

La classe *SimMeasure* est une autre classe importante dans notre modèle de classes. Elle contient les mesures de similarité générées aléatoirement par l'environnement, et permettant aux agents de faire leurs calculs de similarité, lors de la phase de perception. Les mesures de similarité référencées par cette classe, proviennent du référentiel de

mesures de similarité *DKPro*. La sélection aléatoire, par l'agent, des mesures de similarité, tire avantage de l'implémentation des mesures de similarité, du référentiel *DKPro*, d'une interface appelée *TextSimilarityMeasure*.

Touchant à l'aspect dynamique de la modélisation du comportement de l'agent réflexif. D'un côté, le diagramme d'activités (figure 3.5) met en évidence le comportement de l'agent, dont le but est de trouver un appariement consensuel, durant une itération. On peut remarquer, comme le montre ce diagramme, que l'approche de la sélection de l'appariement consensuel est une approche naïve, consistant en l'attente d'un consensus qui doit coïncider, pour les deux agents (ce qui peut impliquer une durée plus longue pour la simulation).

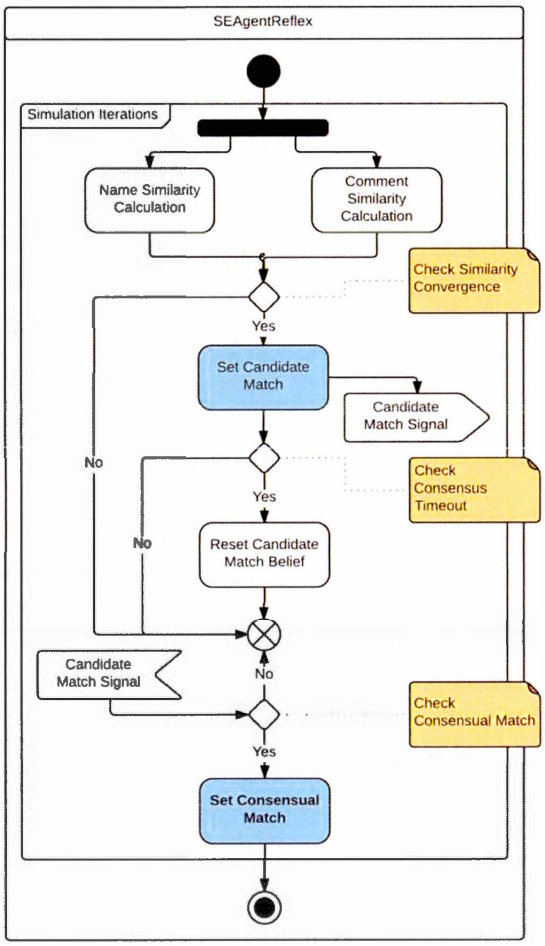


Figure 3.5 Diagramme d'activités du comportement de l'agent *SEAgentReflex*

De l'autre côté, le diagramme d'états-transitions, à base de *machines à états finis* (*FSM*), permet de rendre compte des transitions des états internes, lors de la phase de l'action du cycle *perception-décision-action* de l'agent *SEAgentReflex.*

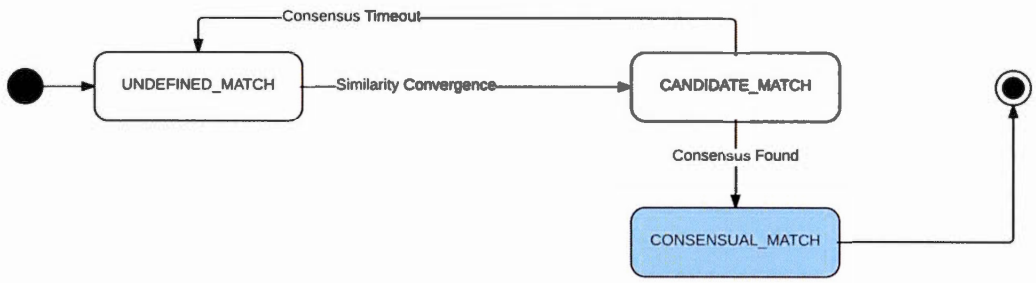


Figure 3.6 Transition entre états de l'agent

Ci-dessous, le pseudocode (squelette du code *Java* de l'agent) décrivant les différentes étapes exécutées par l'agent lors des différentes phases du cycle *perception-décision-action,* d'une itération de simulation.

```
package ca.uqam.latece.smas.reflex.SEAgentReflex;

public class SEAgentReflex {

	/*
	 * Simulation Step 1 - Calculate name similarity: each agent calculates the
	 * similarity between its name and all other agent names (in the other
	 * group) and identifies the best match based on the best name similarity
	 * score. For each similarity calculation a similarity measure is selected
	 * randomly from a similarity measures list. The score should be greater than
	 * a random THRESHOLD (generated random threshold value within interval)
	 */
	@ScheduledMethod(start = 1, interval = 1, priority = 0)
	public void computeNameSim() {
	}

	/*
	 * Simulation Step 2 - Calculate comment similarity: each agent calculates
	 * the similarity between its comment and all other agent comments (in the
	 * other group) and identifies the best match based on the best comment
	 * similarity score. Again, for each similarity calculation a similarity
	 * measure is selected randomly from a similarity measures list. The score
	 * should be greater than a random THRESHOLD (generated random threshold
	 * value within interval)
	 */
	@ScheduledMethod(start = 1, interval = 1, priority = -1)
	public void computeCommentSim() {
	}

	/*
	 * Simulation Step 3 – check for convergence: If the best match for name
	 * and the best match for comment are converging to the same Agent (in the
	 * other group) then the agent identifies it as the candidate match and update
	 * its status from "NO_MATCH" to "CANDIDATE_MATCH". An aggregation
	 * calculation is done between name similarity score and comment similarity
	 * score of the candidate match. The aggregation function is selected
	 * randomly from an aggregation function list (MAX, AVERAGE, WEIGHTED). If
	 * the new aggregated score is grater than a random THRESHOLD (generated
	 * random threshold value within interval) and if it is better that the one
	 * obtained previously the object CandidateMatchingAttribute is updated with
	 * the new scores (name and comment).
	 */
	@ScheduledMethod(start = 1, interval = 1, priority = -2)
	public void checkSimsConvergence() {
	}

	/*
	 * Simulation Step 4 - check for the consensus timeout: If after a certain
	 * number of ticks (e.g. 250) the agent has not yet reached a consensus with
	 * another agent (both are referring to each-other as candidate match) then
	 * the agent beliefs about the candidate match are reset to null and the
	 * state is changed to "NO_MATCH"
	 */
	@ScheduledMethod(start = 1, interval = 1, priority = -3)
	public void checkConsensusTimeout() {
	}

	/*
	 * Simulation Step 5 - wait for Consensus: watch for other agent's candidate
	 * match update in order to check if a consensus was reached with another
	 * agent (both agents are referring to each-other as candidate match). If so
	 * then the agent coalition is updated with the name of the other agent and
	 * the state is changed to "CONSENSUAL_MATCH".
	 */

	@Watch(watcheeClassName = "ca.uqam.latece.smas.agent.reflex.SEAgentReflex",
			watcheeFieldNames = "candidateMatch",
			query = "colocated",
			triggerCondition = "(!$watcher.getAgentGroup().equals($watchee.getAgentGroup()) ) && "
			+ "( $watchee.getCandidateMatch() != null ) && "
			+ "( $watcher.getCandidateMatch() != null ) ",
			whenToTrigger = WatcherTriggerSchedule.IMMEDIATE)
	public void checkConsensus(SEAgentReflex otherAgent) {
	}

	/*
	 * Simulation Step 6 - End the simulation: If all agents have reached
	 * consensus about their matchings then their status is changed to
	 * "STABLE_MATCH" and the simulation is ended.
	 */
	@ScheduledMethod(start = 1, interval = 1, priority = -5)
	public void endSim() {
	}

}
```

3.3.1.1 Implémentation

Notre implémentation du prototype s'est focalisée sur le développement du model opérationnel en gardant manuelles certaines étapes (sans valeurs ajoutées), telles que l'automatisation de l'importation des schémas ou encore l'automatisation de l'exécution des scripts statistique. L'outil *Repast Simphony* offre déjà une interface très riche pour la visualisation, la collecte de données, l'exécution des simulations multiples, etc. La figure 3.7 illustre l'interface utilisateur, basée sur l'interface de l'outil *Repast Simphony*, de notre prototype.

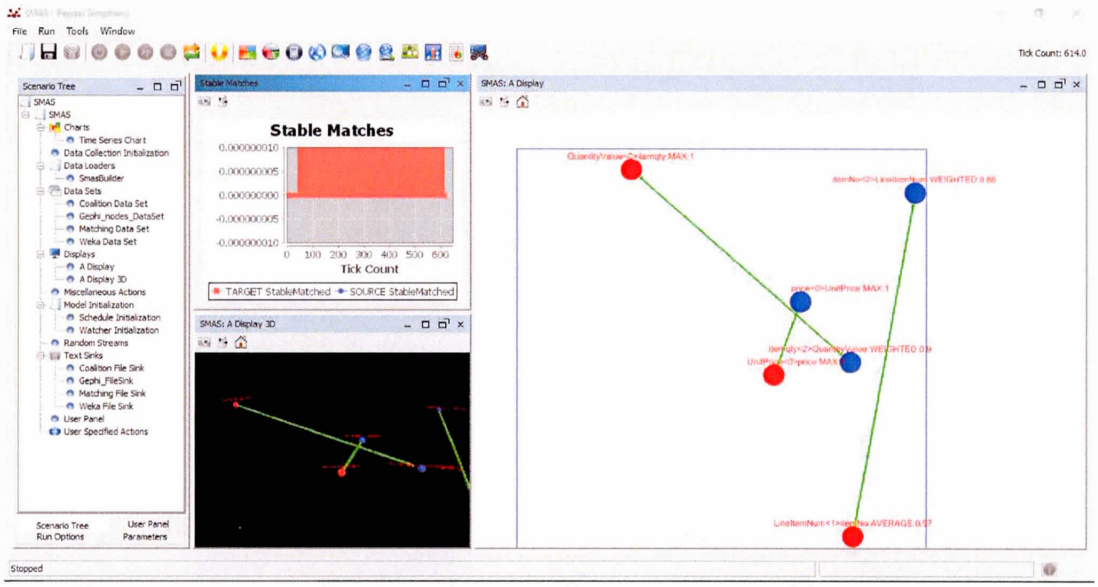


Figure 3.7 Prototype *SMAS* exécuté sur la plateforme *Repast Simphony*

Pendant l'exécution de la simulation, l'utilisateur peut suivre son évolution sur le panneau de visualisation de l'outil *Repast Simphony*. Au début de la simulation, à l'itération 0, comme l'illustre la figure 3.8, le panneau de visualisation nous permet de voir les agents des deux schémas dispersés aléatoirement dans l'espace de l'environnement (contrairement à d'autres modèles de simulation l'emplacement des agents n'a pas d'importance pour notre modèle). Un code de couleur permet de différencier les agents de chaque schéma (la couleur rouge pour les éléments du schéma

source et la couleur bleu pour les éléments du schéma cible). On peut remarquer aussi, que les agents affichent, comme libellé, l'attribut *name* (i.e. nom de l'élément de schéma).

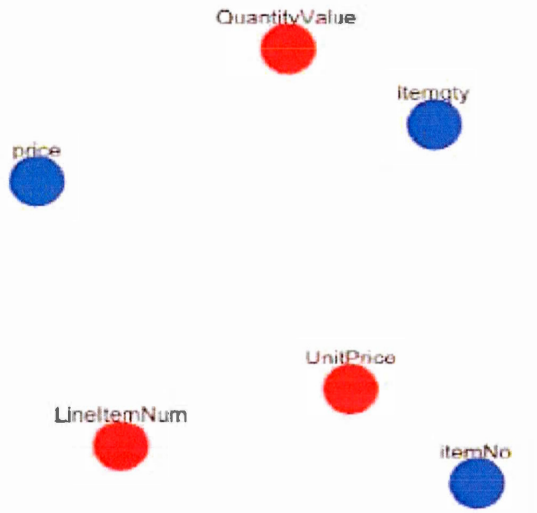


Figure 3.8 Panneau de visualisation au début de la simulation

Durant la simulation, des arcs dirigés avec respectivement la couleur grise pour un appariement candidat et la couleur verte pour un appariement consensuel, sont créés entre les agents. Lorsqu'un consensus est trouvé entre deux agents, le libellé de chaque agent, de la paire formant le consensus, affiche quelques informations, concernant la trajectoire ayant menée l'agent à trouver l'appariement consensuel (une concaténation du nom de l'agent, la fonction agrégation, le score agrégé, ainsi que le nom de l'autre agent de la paire). La figure 3.9 illustre la visualisation de la fin d'une simulation.

Figure 3.9 Panneau de visualisation à la fin de la simulation

Concernant les paramètres de configuration, nous avons choisi sciemment de ne pas en avoir, si l'on exclut le paramètre permettant de contrôler la durée de la simulation (i.e. nombre d'itérations). Ce choix est justifié, d'une part, par le fait que toutes les valeurs des paramètres importants pour l'appariement sont générées aléatoirement, par l'environnement durant la simulation, et d'autre part, par le fait que nous voulions, en adoptant ce choix, mettre en relief, sans ambiguïté aucune, la nature autonome et adaptative de notre approche.

3.3.2 Expérimentations

Au-delà des justifications théoriques pour le choix de la simulation multi-agents pour la résolution de l'appariement de schémas, et à mesure que les premières expérimentations sur notre prototype se multipliaient, une citation de (Schruben, 2015) concernant la puissance virtuelle illimitée de la simulation, venait résumer d'une façon à la fois simple et percutante nos premières impressions : « *Simulation models provide virtually unlimited power; or rather, they provide unlimited virtual power. If you can*

think of something, you can simulate it. Experimenting in a simulated world, you can change anything, in any way, at any time - even change time itself. ».

En ce qui a trait à l'utilisation de la simulation dans le contexte de la problématique de l'appariement, les premières expérimentations nous ont démontré à quel point la simulation était un outil puissant pour donner des réponses, d'une manière naturelle, à des questions tombant dans la catégorie de *l'analyse d'hypothèses*[82]. Des questions qui s'intéressent à ce qui pourrait se produire, dans le cadre d'un scénario d'appariement donné, avec une combinaison donnée, d'apparieurs, de fonctions d'agrégation, de paramètres de configuration, plutôt qu'avec une autre. Le nombre exponentiel des cas, qui peut conduire à la problématique de l'explosion combinatoire, qui constitue un frein pour les approches classiques d'appariement, se retrouve contourné d'une façon élégante par l'approche de simulation.

L'approche adoptée, pour l'expérimentation de notre prototype, se décompose en 2 étapes principales :

1. Identification d'un point de référence : cette étape est consacrée à l'identification d'un scénario d'appariement et à l'expérimentation de ce scénario d'appariement avec un outil connu de la littérature, et ce dans le but, d'un côté, d'en évaluer la pertinence, et de l'autre, d'avoir des résultats pouvant être considérés comme un point de référence[83] pour les expérimentations futures avec notre prototype.
2. Expérimentation avec le prototype : cette étape est consacrée à l'évaluation préliminaire des performances de notre prototype, d'un côté, au regard des

[82] De l'anglais « *what-if analysis* »

[83] De l'anglais « *baseline* »

résultats attendus par l'utilisateur, et de l'autre côté, au regard des résultats obtenus comme point de référence.

Les décisions concernant l'approche de l'expérimentation adoptée, peuvent se décliner comme suit :

- La première décision concerne le choix de l'exemple de référence (chapitre 2) comme scénario d'appariement pour les expérimentations préliminaires. Le critère pour le choix du scénario peut se résumer ainsi : le scénario doit être à la fois facile à décrire, pour mieux expliquer le déroulement des expérimentations, et poser un certain niveau de difficulté pour mieux situer les performances obtenues lors des expérimentations. L'exemple de référence est un scénario assez simple avec deux schémas ayant peu d'éléments scénario (i.e. un exemple jouet[84] de 3 éléments). Mais est-il assez difficile ? Pour répondre à cette question, nous avons donc décidé d'en évaluer la pertinence en le testant avec un outil bien connu dans la littérature.
- La seconde décision concerne le choix de l'outil *COMA* comme outil nous permettant d'avoir une évaluation empirique sommaire, de la difficulté que peut poser ce scénario d'appariement à un outil d'appariement automatique de schémas reconnu, avec l'optique de garder les résultats qui découleront de ce test comme un référentiel.

Il est très important de préciser que ce test, ne se veut en aucun cas être une comparaison des performances (i.e. étude comparative[85]) de notre prototype par rapport aux autres outils existants, et encore moins par rapport à l'outil *COMA*.

---

[84] De l'anglais « *toy example* »

[85] De l'anglais « *benchmarking* »

Avant d'aller plus loin, il convient de rappeler l'exemple de référence, choisi comme scénario d'appariement pour l'expérimentations de notre prototype, à travers les résultats attendus par l'utilisateur. Les correspondances à trouver sont comme suit :

- *LineItemNum↔itemNo*
- *UnitPrice↔price*
- *QuantityValue↔itemqty*

Même s'il ne s'agit nullement d'un scénario d'appariement très complexe (i.e. deux schémas en entrée avec 3 éléments), et après son évaluation avec l'outil *COMA*, le résultat obtenu, montre clairement que cet exemple pourrait être adopté pour l'expérimentation de notre prototype dans la mesure où il peut représenter un défi intéressant. Deux observations majeures ont pu ressortir de ce test. La première observation est que l'outil *COMA*, avec son processus d'appariement utilisant des stratégies (i.e. flux de travail) par défaut (i.e. *$NodesNameS* et *$NodesS*), n'était pas capable de trouver toutes les correspondances attendues par l'utilisateur. La seconde observation est que la répétition de ce test, sans changement de configuration, conduisait toujours au même résultat.

Comme illustré par la figure 3.10, les deux premières correspondances (i.e. *LineItemNum↔itemNo* et *UnitPrice↔price*) ont été trouvées avec succès (en utilisant les flux de travail *$NodesS*). En revanche, l'outil a été incapable de trouver la troisième correspondance attendue (i.e. *QuantityValue↔itemqty*). Du reste, l'utilisation d'un autre flux de travail de *COMA* (i.e. *$NodesNameS*) n'a pas permis de donner de meilleurs résultats, mais au contraire, dans certains cas, l'outil n'a été capable de trouver qu'une seule correspondance (i.e. *UnitPrice↔price*) (une correspondance sur trois).

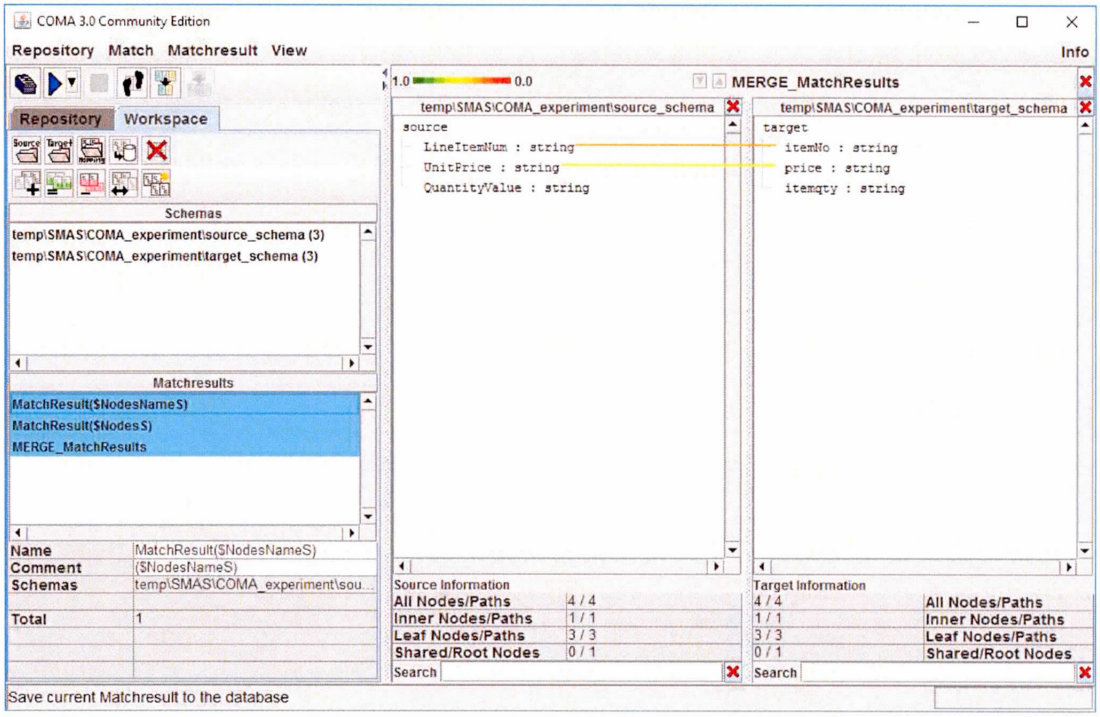


Figure 3.10 Test du scénario « *exemple de référence* » avec l'outil *COMA*

Une union des résultats des deux flux de travail, *$NodesNameS* et *$NodesS*, a été faite avec l'outil *COMA* pour générer un résultat *MERGE_MatchResults* qui contient la sélection des correspondances ayant le meilleur score.

Il est essentiel de préciser que les deux workflow (stratégies d'appariement) *$NodesNameS* et *$NodesS* (tenant compte uniquement des éléments de schémas et non pas de la structure des schémas) ont été choisis comme stratégie d'appariement pour ce test, dans un souci de rapprochement, le plus possible, de la stratégie d'appariement de notre prototype, qui dans sa version actuelle ne tient pas compte de la structure des schémas en entrée.

Ci-dessous, l'export du résultat de ce scénario d'appariement obtenu avec *COMA*.

```
MatchResult($NodesNameS) [4,4]
-------------------------------------------------------
 - 368[UnitPrice:string] <-> 373[price:string]: 0.555556
 + Total: 1 correspondences
-------------------------------------------------------

MatchResult($NodesS) [4,4]
-------------------------------------------------------
 - 367[LineItemNum:string] <-> 372[itemNo:string]: 0.297763
 - 368[UnitPrice:string] <-> 373[price:string]: 0.486652
 + Total: 2 correspondences
-------------------------------------------------------

MERGE_MatchResults [4,4]
-------------------------------------------------------
 - 367[LineItemNum:string] <-> 372[itemNo:string]: 0.297763
 - 368[UnitPrice:string] <-> 373[price:string]: 0.555556
 + Total: 2 correspondences
-------------------------------------------------------
```

Notons au passage que l'expérimentation tient aussi compte du fait que l'on veut comparer la facilité d'utilisation. Donc nous n'avons pas chercher à utiliser des options avancées de l'outil, ce qui aurait permis peut-être de trouver la dernière correspondance.

Donc, maintenant, le défi pour notre prototype est clair : faire au moins aussi bien si non mieux : 100% des correspondances attendues trouvées.

3.3.2.1 Déroulement de la simulation

Afin de mieux illustrer le déroulement du modèle opérationnel de notre simulation multi-agents pour l'appariement de schémas, nous allons pendant une simulation donnée, effectuer des pauses à différents moments. Ces pauses vont permettre la capture de l'état de la simulation à ces moments. Pour se faire, nous allons procéder à l'interrogation de l'état interne des agents (quelques attributs importants) et ce en

utilisant la librairie *JoSQL*[86] (Bentley, s.d.) (*panneau SQL*), enchâssée dans la plateforme *Repast Simphony*.

La figure 3.11 montre le panneau *SQL* (i.e. interface *JoSQL*) pour l'interrogation de l'état des agents à un moment donnée de la simulation.

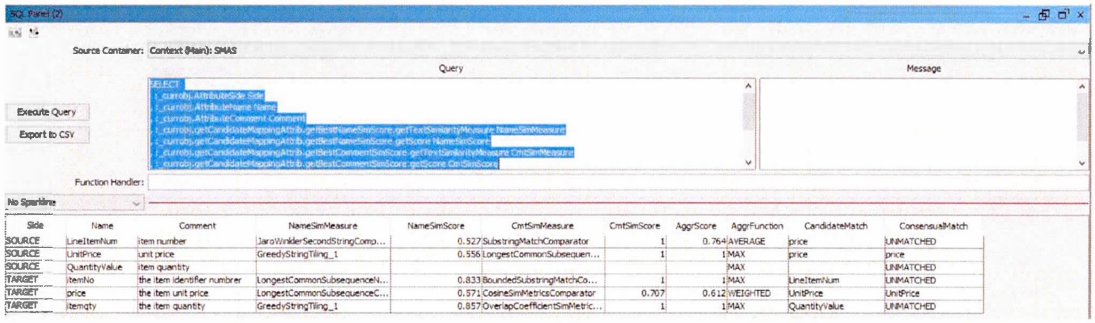


Figure 3.11 Interface *JoSQL* pour l'interrogation de l'état de la simulation

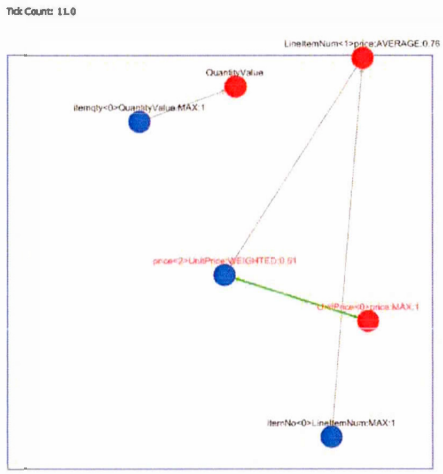
Ci-dessous, un exemple de code *JoSQL* permettant l'interrogation des attributs des agents lors de la simulation.

```
SELECT
  :_currobj.AttributeSide Side
, :_currobj.AttributeName Name
, :_currobj.AttributeComment Comment
, :_currobj.getCandidateMappingAttrib.getBestNameSimScore.getTextSimilarityMeasure NameSimMeasure
, :_currobj.getCandidateMappingAttrib.getBestNameSimScore.getScore NameSimScore
, :_currobj.getCandidateMappingAttrib.getBestCommentSimScore.getTextSimilarityMeasure CmtSimMea-
sure
, :_currobj.getCandidateMappingAttrib.getBestCommentSimScore.getScore CmtSimScore
, :_currobj.getCandidateMappingAttrib.getBestSimScore.getScore AggrScore
, :_currobj.getAggregationFunctionName AggrFunction
, :_currobj.getCandidateMappingAttrib.getMatchedAttributeName CandidateMatch
,:_currobj.candidateCoallitionAttrib ConsensualMatch
FROM java.lang.Object
```

Une série de clichés ont été pris à des moments différents de la simulation, nous renseignant ainsi sur l'évolution et la coévolution (*agent-agent* et *agent-*

[86] SQL for Java Objects (JoSQL) library

environnement). La figure 3.12 nous montre, par exemple, la situation à l'itération 11 de la simulation



Side	Name	Comment	NameSimMeasure	NameSimScore	CmtSimMeasure	CmtSimScore	AggrScore	AggrFunction	CandidateMatch	ConsensualMatch
SOURCE	LineItemNum	item number	JaroWinklerSecondStringComp...	0.527	SubstringMatchComparator	1	0.764	AVERAGE	price	UNMATCHED
SOURCE	UnitPrice	unit price	GreedyStringTiling_1	0.556	LongestCommonSubsequen...	1	1	MAX	price	price
SOURCE	QuantityValue	item quantity						MAX		UNMATCHED
TARGET	itemNo	the item identifier number	LongestCommonSubsequenceN...	0.833	BoundedSubstringMatchCo...	1	1	MAX	LineItemNum	UNMATCHED
TARGET	price	the item unit price	LongestCommonSubsequenceC...	0.571	CosineSimMetricsComparator	0.707	0.612	WEIGHTED	UnitPrice	UnitPrice
TARGET	itemqty	the item quantity	GreedyStringTiling_1	0.857	OverlapCoefficientSimMetric...	1	1	MAX	QuantityValue	UNMATCHED

Figure 3.12 Aperçu de la simulation a l'itération 11

La situation concernant l'appariement à l'itération 11 est la suivante :

- Les agents *UnitPrice* et *price* ont déjà formé un appariement consensuel stable
- L'agent *itemqty* est sur la bonne voie en choisissant l'agent *QuantityValue* comme appariement candidat
- L'agent *itemNo* lui aussi est sur la bonne voie en choisissant l'agent *LineItemNum* comme appariement candidat
- L'agent *LineItemNum* quant à lui fait un choix erroné en choisissant l'agent *price* comme appariement candidat
- L'agent *QuantityValue* n'a pas encore trouvé un appariement candidat

À l'itération 195 nous remarquons, par rapport à la situation précédente, que :

- L'agent *LineItemNum* qui avait fait un choix erroné efface ses croyances concernant le choix de l'agent *price* comme appariement candidat

À l'itération 202 nous remarquons, par rapport à la situation précédente, que :

- Les agents *itemNo* et *LineItemNum* ont réussi à former eux aussi un appariement consensuel et stable

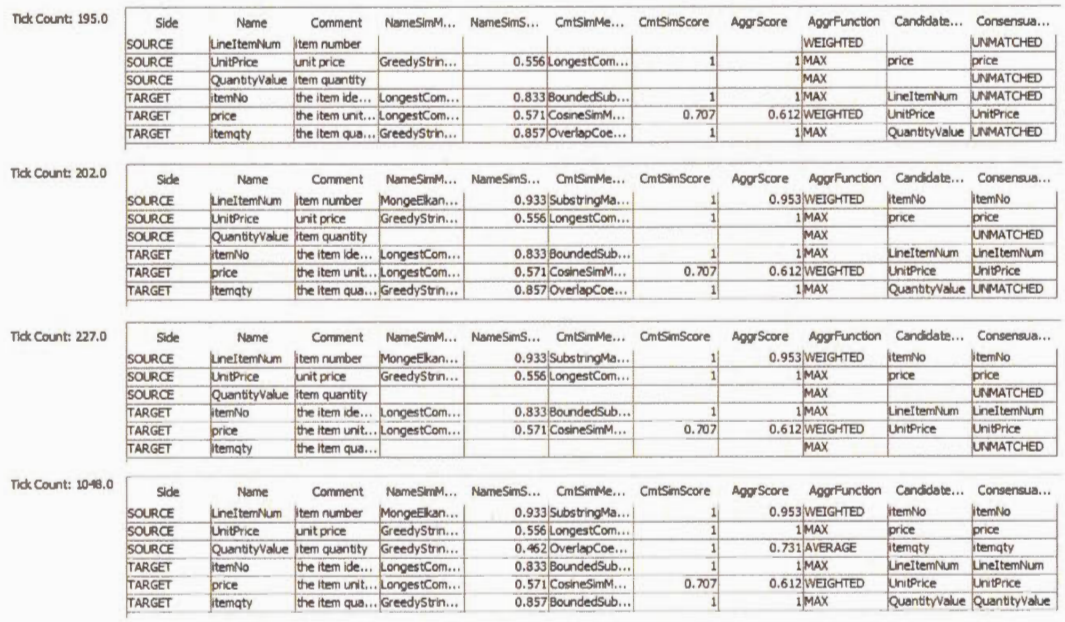
À l'itération 227 nous remarquons, par rapport à la situation précédente, que :

- L'agent *itemqty* qui était sur la bonne voie, en choisissant l'agent *QuantityValue* comme appariement candidat. Cependant, après une longue attente d'un consensus (qu'il soit aussi sélectionné par *QuantityValue*), l'agent *itemqty* décide d'effacer ses croyances concernant le choix de l'agent *QuantityValue* comme appariement candidat

Enfin à la fin de la simulation, c'est-à-dire à l'itération 1048, nous remarquons que :

- Les agents *itemqty* et *QuantityValue*, au bout de quelques tentatives infructueuses d'arriver en même temps à un consensus (respectivement 6 et 7 opérations d'effacement de croyances concernant l'appariement candidat), ont fini par avoir un consensus et ainsi former le dernier appariement consensuel, signalant par la même occasion la fin de la simulation.

Il est opportun de noter que les 1048 itérations qui étaient nécessaires lors cette simulation, pour trouver le résultat, ont duré un peu moins de trois secondes.



Tick Count: 195.0

Side	Name	Comment	NameSimM...	NameSimS...	CmtSimMe...	CmtSimScore	AggrScore	AggrFunction	Candidate...	Consensua...
SOURCE	LineItemNum	item number						WEIGHTED		UNMATCHED
SOURCE	UnitPrice	unit price	GreedyStrin...	0.556	LongestCom...	1	1	MAX	price	price
SOURCE	QuantityValue	item quantity						MAX		UNMATCHED
TARGET	itemNo	the item ide...	LongestCom...	0.833	BoundedSub...	1	1	MAX	LineItemNum	UNMATCHED
TARGET	price	the item unit...	LongestCom...	0.571	CosineSimM...	0.707	0.612	WEIGHTED	UnitPrice	UnitPrice
TARGET	itemqty	the item qua...	GreedyStrin...	0.857	OverlapCoe...	1	1	MAX	QuantityValue	UNMATCHED

Tick Count: 202.0

Side	Name	Comment	NameSimM...	NameSimS...	CmtSimMe...	CmtSimScore	AggrScore	AggrFunction	Candidate...	Consensua...
SOURCE	LineItemNum	item number	MongeElkan...	0.933	SubstringMa...	1	0.953	WEIGHTED	itemNo	itemNo
SOURCE	UnitPrice	unit price	GreedyStrin...	0.556	LongestCom...	1	1	MAX	price	price
SOURCE	QuantityValue	item quantity						MAX		UNMATCHED
TARGET	itemNo	the item ide...	LongestCom...	0.833	BoundedSub...	1	1	MAX	LineItemNum	LineItemNum
TARGET	price	the item unit...	LongestCom...	0.571	CosineSimM...	0.707	0.612	WEIGHTED	UnitPrice	UnitPrice
TARGET	itemqty	the item qua...	GreedyStrin...	0.857	OverlapCoe...	1	1	MAX	QuantityValue	UNMATCHED

Tick Count: 227.0

Side	Name	Comment	NameSimM...	NameSimS...	CmtSimMe...	CmtSimScore	AggrScore	AggrFunction	Candidate...	Consensua...
SOURCE	LineItemNum	item number	MongeElkan...	0.933	SubstringMa...	1	0.953	WEIGHTED	itemNo	itemNo
SOURCE	UnitPrice	unit price	GreedyStrin...	0.556	LongestCom...	1	1	MAX	price	price
SOURCE	QuantityValue	item quantity						MAX		UNMATCHED
TARGET	itemNo	the item ide...	LongestCom...	0.833	BoundedSub...	1	1	MAX	LineItemNum	LineItemNum
TARGET	price	the item unit...	LongestCom...	0.571	CosineSimM...	0.707	0.612	WEIGHTED	UnitPrice	UnitPrice
TARGET	itemqty	the item qua...						MAX		UNMATCHED

Tick Count: 1048.0

Side	Name	Comment	NameSimM...	NameSimS...	CmtSimMe...	CmtSimScore	AggrScore	AggrFunction	Candidate...	Consensua...
SOURCE	LineItemNum	item number	MongeElkan...	0.933	SubstringMa...	1	0.953	WEIGHTED	itemNo	itemNo
SOURCE	UnitPrice	unit price	GreedyStrin...	0.556	LongestCom...	1	1	MAX	price	price
SOURCE	QuantityValue	item quantity	GreedyStrin...	0.462	OverlapCoe...	1	0.731	AVERAGE	itemqty	itemqty
TARGET	itemNo	the item ide...	LongestCom...	0.833	BoundedSub...	1	1	MAX	LineItemNum	LineItemNum
TARGET	price	the item unit...	LongestCom...	0.571	CosineSimM...	0.707	0.612	WEIGHTED	UnitPrice	UnitPrice
TARGET	itemqty	the item qua...	GreedyStrin...	0.857	BoundedSub...	1	1	MAX	QuantityValue	QuantityValue

Figure 3.13 Différents aperçus de la simulation

La figure ci-dessus permet d'avoir une meilleure idée sur l'évolution de la simulation, en mettant côte à côte le résultat de différents aperçus. Deux observations peuvent être faite :

- Les mesures sont différentes pour chaque agent et lors de chaque pause (sélection aléatoire des mesures de similarité)
- Les agents, d'une manière autonome et en parallèle, empruntent des chemins différents pour arriver à leur but.

Nous pouvons clairement voir, par exemple, que sous l'effet de la sélection aléatoire des mesures de similarité, ces dernières ne sont pas seulement différentes (colonnes *NameSimMeasure* et *CmtSimMeasure*) pour l'attribut *name* ou pour l'attribut *comment* (*CmtSimMeasure*), ou encore d'un agent à l'autre, mais encore elles sont différentes même pour deux agents formant un appariement consensuel (i.e. *UnitPrice*↔*price*).

La même observation peut être faite concernant les fonctions d'agrégation (la colonne *AggrFunction*).

Il en résulte que malgré les divergences qui marquent les trajectoires empruntées par chaque agent, pour arriver à un appariement consensuel, il n'en demeure pas moins que les agents arrivent à former les correspondances attendues par l'utilisateur.

La figure 3.14 montre le résultat final de la simulation tel qu'il apparait dans le panneau de visualisation de l'outil *Repast Simphony*.

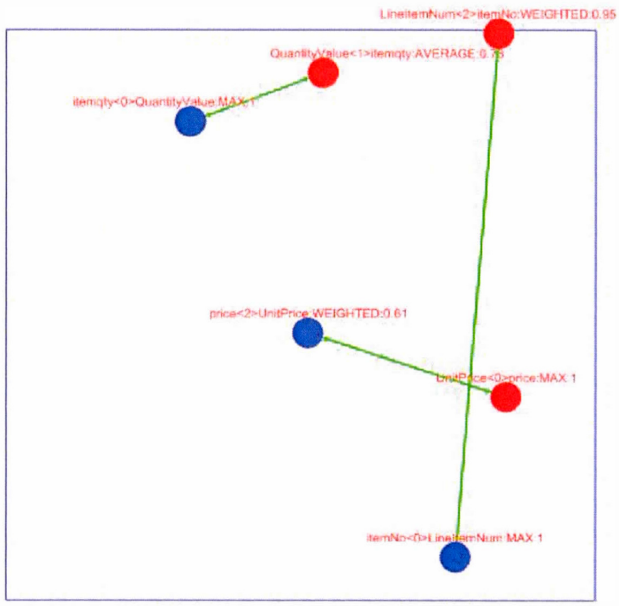


Figure 3.14 Visualisation du résultat final (itération 1048)

Nous estimons devoir insister sur le fait que cette simulation a été exécutée sans aucune configuration préalable, ni aucune optimisation tel que le choix préalable de mesures selon le scénario, le calibrage des paramètres (e.g. le seuil, le poids) ou encore le choix des fonctions, etc.

Au lieu de la configuration et l'optimisation, nous avons pu voir à l'œuvre, tout au long de la simulation dans une dimension temporelle, le mécanisme de la *coévolution* contraignant les agents, *de facto,* à la fois à une configuration et une optimisation *ad-lib* (se répétant plusieurs fois) et *ad-hoc* (sur mesure à la situation). Cette *coévolution* va favoriser le mécanisme *d'auto-organisation* grâce auquel *l'émergence* de la solution finale survient.

Cette précision prend tout son sens lorsque l'on se remémore tout ce qui est écrit dans la littérature sur le défi que représente la configuration et l'optimisation dans l'appariement automatique de schémas. En effet, l'expérimentation faite sur notre prototype, vient de prouver qu'il est possible de trouver des résultats corrects, et ce, sans aucune préparation à priori (qui ne requiert de l'optimisation). Ce résultat vient contraster fortement avec l'idée « fataliste » qui prédomine dans la littérature et qui veut que l'optimisation du processus d'appariement est une tâche inéluctable, difficile et fastidieuse (que ça soit manuellement ou à l'aide d'un processus automatique), comme le souligne par exemple (Sayyadian *et al.*, 2005), dans leur publication sur l'optimisation de l'appariement de schémas : « *...While valuable, tuning is also very difficult, due to the large number of knobs involved, the wide variety of matching techniques employed (e.g., database, machine learning, IR, information theory, etc.), and complex interaction among the components. Writing a "user manual" for tuning seems nearly impossible...matching systems are still tuned manually, largely by trial and error – a time consuming, frustrating, and error prone process...* ».

3.3.2.2 Discussion des résultats

Les résultats obtenus, sur plusieurs exécutions de la simulation, démontrent clairement que notre prototype réussi dans la majorité des cas (nature non déterministe contrairement aux outils classiques) à trouver les correspondances attendues par l'utilisateur (i.e. *LineItemNum*↔*itemNo*, UnitPrice↔*price* et *QuantityValue*↔*itemqty*). Cela prouve que notre prototype arrive avec cet exemple

jouet à tirer son épingle du jeu en faisant mieux que l'outil *COMA* qui, à titre de rappel, n'a pas pu trouver la correspondance (i.e. *QuantityValue↔itemqty*), et ce malgré la répétition des tests (nature déterministe). Le haut degré d'ambigüité lexicale (pas beaucoup de ressemblance au niveau des noms de schémas) est sans doute la cause de cette échec à trouver la troisième correspondance de la part de *COMA*.

Il faut préciser toutefois, qu'à ce stade-ci nous nous gardons de tirer toute conclusion hâtive ou de tomber dans des comparaisons non fondées sur un processus empirique rigoureux. Le but visé était de démontrer, que malgré la simplicité apparente de l'exemple jouet, il reste quand même, grâce à l'aspect ambigüe de ses éléments de schéma (comme le prouve le résultat de l'outil *COMA*), un bon scénario pour l'évaluation préliminaire de la performance de notre prototype.

Un autre point qui a attiré notre attention est celui du nombre de calculs de similarité réalisés durant une simulation. La figure 3.15 montre un graphe indiquant le compte des calculs de similarité, effectué par les agents, par rapport au nombre des itérations lors d'une simulation donnée. On peut voir, par exemple, lors de cette simulation que la durée totale a été de 52 itérations. Durant ces itérations, le nombre total de fois qu'un calcul de similarité a été effectué, par tous les agents, et représenté par la courbe de couleur bleu, approche les 1000 fois. Les courbes de couleur rouge et verte représentent quant à elles, le nombre de calcul de similarité ayant conduit à un score supérieur au seuil (aléatoire) pour, respectivement, l'attribut *name* approche les 50 alors que pour l'attribut *comment* le nombre total approche les 200.

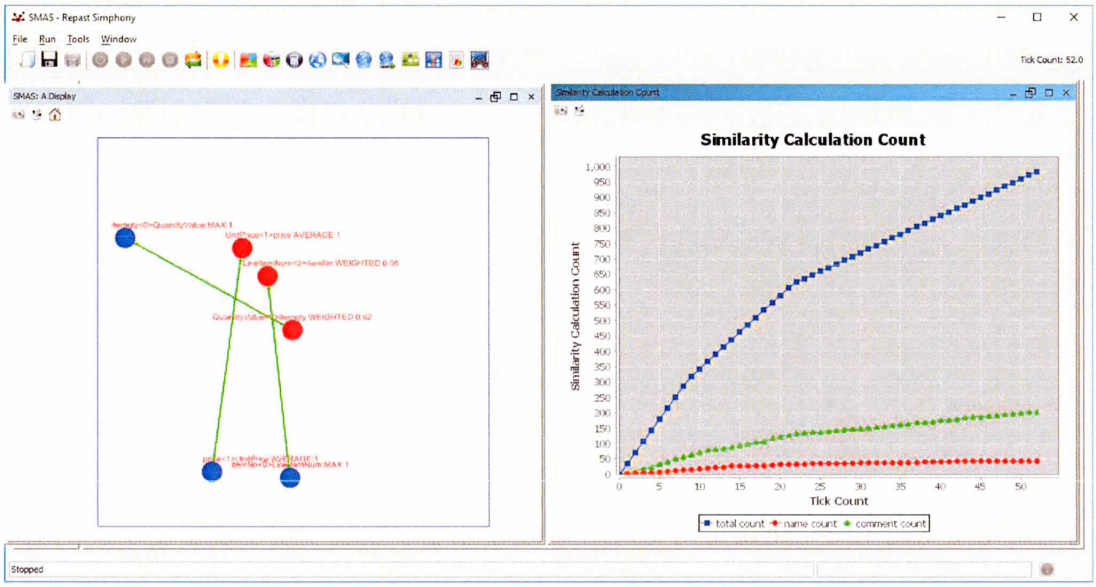


Figure 3.15 Nombre de calculs de similarité lors d'une simulation

Tout d'abord, en examinant les chiffres concernant les calculs de similarité durant cette simulation, on constate le très grand nombre de calculs de similarité effectué par les agents durant une simulation avec une durée de vie relativement courte (936 calculs de similarité durant 52 itérations). Ensuite, on constate, que le nombre de calculs de similarité dépassant le seuil, pour l'attribut *name,* est inférieur à celui de l'attribut *comment*. L'explication pourrait résider dans le fait que les mesures de similarité utilisées lors de cette simulation, accordent, sans doute des scores plus élevés lorsqu'il s'agit de chaine de mots (i.e. attribut *comment*), que lorsqu'il s'agit de chaines de caractères (i.e. attribut *name*).

$$\text{tot. similarity calc.} = \text{agents} \times \sum_{i=1}^{\text{iterations}} (\text{similarity calc.} \times \text{other agents})$$

(3.5) Formule pour le nombre total des calculs de similarité lors d'une simulation

Par exemple, si nous appliquons la formule sur le scénario précèdent, avec *agent* = 3, *other agents* = 3, *similarity calc.* = 2 (calcul de similarité pour le nom et pour le commentaire) et enfin *itérations* = 52 on obtient :

$$\text{tot. similarity calc.} = 936 = 3 \times \sum_{i=1}^{52} (2 \times 3)$$

L'examen des expérimentations effectuées sur cette première version du prototype (version réflexive) démontre que le prototype non seulement arrive à trouver les résultats attendus par l'utilisateur, mais plus il arrive à produire ces résultats en matérialisant les concepts théoriques avancés plus tôt (chapitre 2), à savoir : autonomie, décentralisation, stochasticité, etc. On convient toutefois que le modèle réactif reste un modèle très naïf.

En conclusion, la construction du prototype, la conduite des expérimentations et l'analyse des résultats sont autant d'étapes franchies sur le chemin de la validation de notre postulat de base, à savoir, considérer l'appariement automatique de schémas comme un système complexe adaptatif. En effet, les expérimentations réalisées sur notre prototype nous poussent à confirmer, même avec prudence, que l'idée qui consiste à dire que l'appariement de schémas peut être modélisé et résolu avec une simulation multi-agents est une idée viable.

3.3.2.3 Simulations multiples et analyse statistique

Après de nombreuses exécutions individuelles de la simulation, on remarque que le prototype, grâce à sa nature non-déterministe, et grâce à la part que prend l'aléatoire dans notre modèle (contrairement aux outils classiques), réussi, certes, à trouver 100% des correspondances attendues, mais non pas dans 100% des exécutions. En d'autre mots, avec les mêmes conditions initiales (i.e. même scénario d'appariement), dans la majorité des cas on arrive à avoir les correspondances attendues (avec à chaque fois

des trajectoires différentes pour les agents). En revanche dans certains autres cas, le prototype n'arrive pas à trouver 100% des correspondances. Cette remarque dénote du caractère non-prévisible de notre outil, découlant de ses caractéristiques intrinsèquement fondamentales.

Avec la vérification empirique, en conduisant des expérimentations sur notre prototype du modèle opérationnel, nous avons par la même occasion pu observer la manifestation de pratiquement tous les aspects théoriques (avancés dans le chapitre 2) qui singularisent notre modèle conceptuel, à savoir *la stochasticité et l'adaptation* pour le calcul de similarité, *l'appariement consensuel* pour la sélection des correspondances et *l'auto-organisation et l'émergence* de l'alignement. Le seul aspect dont nous n'avons pas encore pu voir la manifestation est celui, visant la quantification de l'incertitude, c'est à dire celui des *simulations multiples et l'analyse statistique*. La raison à cela, et que cet aspect, comme son nom l'indique, ne peut se manifester qu'à travers l'exécution d'une *méta-simulation* (simulations multiples) et non pas à travers des exécutions individuelles et isolées de la simulation.

Dans cette section, nous allons démontrer empiriquement, pourquoi la variabilité, somme toute réduite, des résultats (non-prévisibles, non-linéaires, bruitées et non-monotones), selon les exécutions de la simulation, constitue pour nous un avantage, et paradoxalement un pas vers la réduction de l'incertitude, et comment les simulations multiples et l'analyse statistique permettent de mieux quantifier l'incertitude (en exploitant la fréquence « *loi des grands nombres* »).

Notre expérimentation de l'aspect concernant les simulations multiples va tirer avantage d'une fonctionnalité pratique de l'outil *Repast Simphony,* à savoir la fonctionnalité « exécution en lot[87] » (la figure 3.16 montre l'interface d'exécution en lot de l'outil *Repast Simphony*). En effet, l'outil *Repast Simphony* permet une exécution

[87] De l'anglais « *batch run* »

individuelle de la simulation (panneau de visualisation), ou encore une exécution de celle-ci en lot. En revanche, lorsqu'il s'agit d'une exécution en lot, la visualisation du déroulement de la simulation n'est plus possible. Le déroulement des différentes simulations du lot, n'est pas pour autant opaque à l'utilisateur, il peut toujours être capturé dans des fichiers texte.

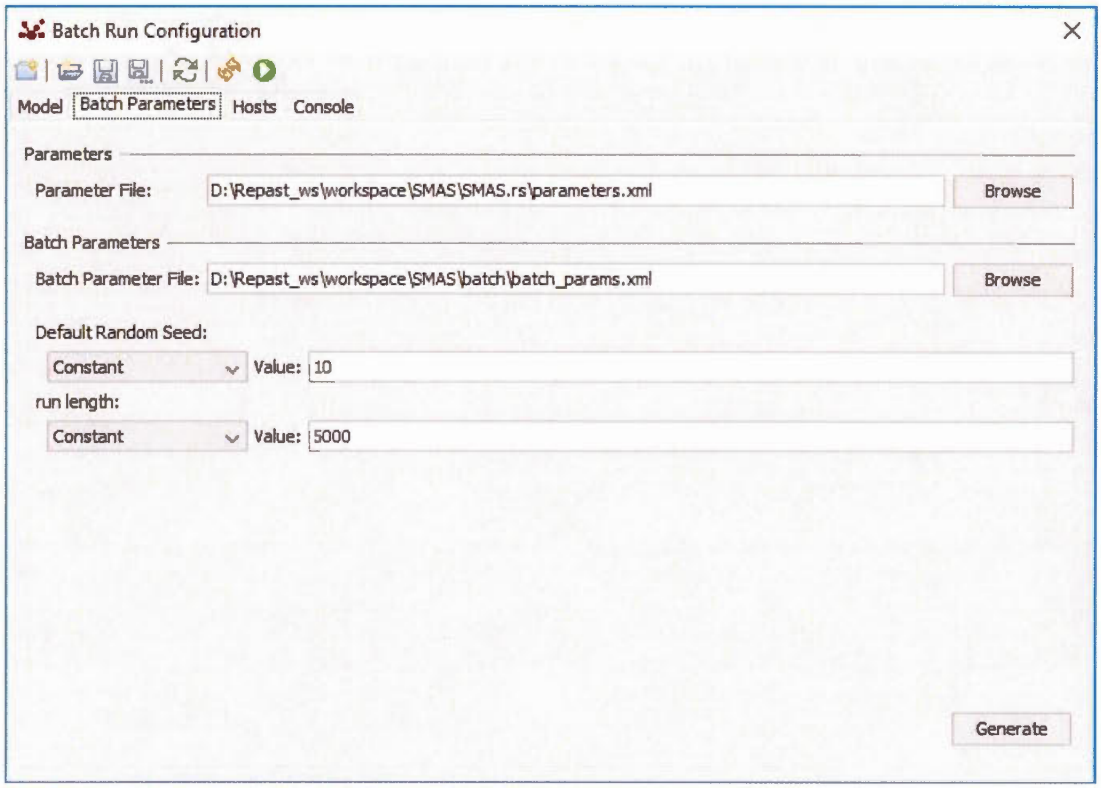
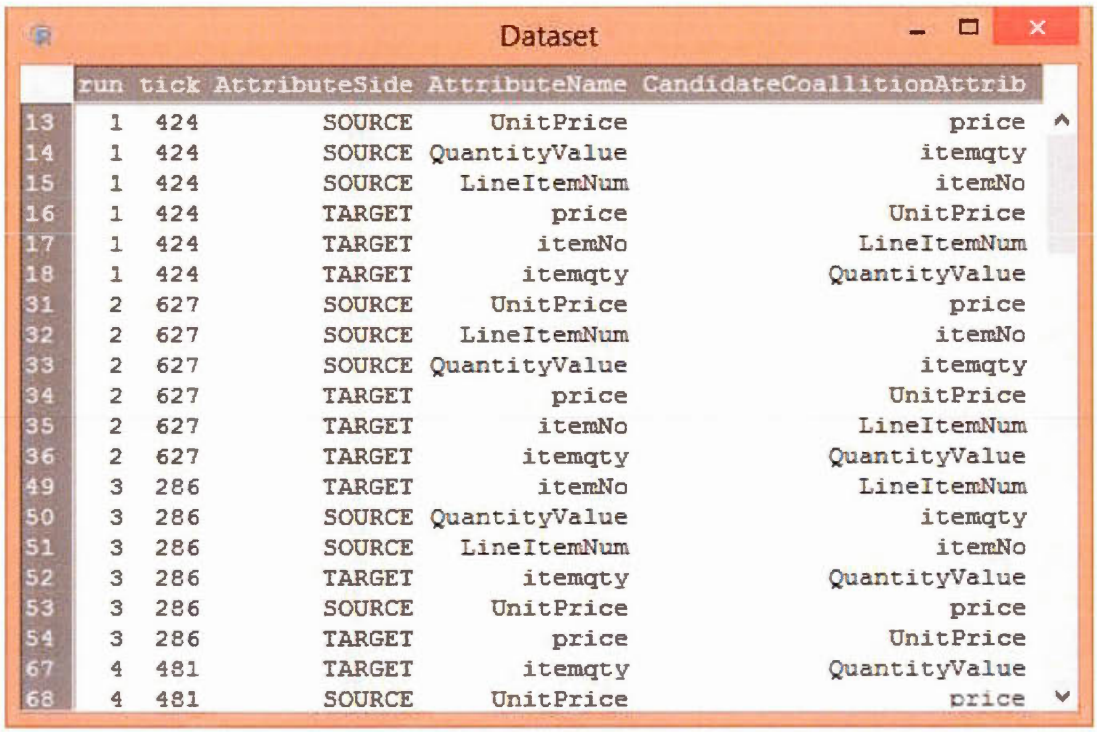


Figure 3.16 Interface d'exécution des simulations en lot

La figure 3.17 montre une partie du résultat final d'une exécution en lot de notre modèle opérationnel. On peut remarquer que pour chaque simulation, on a son numéro avec le numéro d'exécution (colonne *run*), sa durée de vie avec le nombre d'itérations[88] (colonne *tick*), le groupe de l'agent (schéma en entrée), et enfin la paire d'agents ayant

[88] De l'anglais (*Tick*)

formé un appariement consensuel. Notons au passage, la variabilité qui caractérise la durée de vie de chaque simulation (*tick*). En effet, comme chaque simulation est différente, le temps nécessaire pour trouver les correspondances varie d'une simulation à l'autre.



	run	tick	AttributeSide	AttributeName	CandidateCoallitionAttrib
13	1	424	SOURCE	UnitPrice	price
14	1	424	SOURCE	QuantityValue	itemqty
15	1	424	SOURCE	LineItemNum	itemNo
16	1	424	TARGET	price	UnitPrice
17	1	424	TARGET	itemNo	LineItemNum
18	1	424	TARGET	itemqty	QuantityValue
31	2	627	SOURCE	UnitPrice	price
32	2	627	SOURCE	LineItemNum	itemNo
33	2	627	SOURCE	QuantityValue	itemqty
34	2	627	TARGET	price	UnitPrice
35	2	627	TARGET	itemNo	LineItemNum
36	2	627	TARGET	itemqty	QuantityValue
49	3	286	TARGET	itemNo	LineItemNum
50	3	286	SOURCE	QuantityValue	itemqty
51	3	286	SOURCE	LineItemNum	itemNo
52	3	286	TARGET	itemqty	QuantityValue
53	3	286	SOURCE	UnitPrice	price
54	3	286	TARGET	price	UnitPrice
67	4	481	TARGET	itemqty	QuantityValue
68	4	481	SOURCE	UnitPrice	price

Figure 3.17 Résultat de multiples simulations

Une analyse statistique est ensuite faite sur le résultat final de multiples simulations (15 simulations) d'appariement à base d'agents. La figure 3.18 illustre comment le résultat des simulations est synthétisé sous la forme d'un tableau de contingence[89] (*matrice des correspondances*).

[89] De l'anglais « *contingency table* »

	CandidateCoallitionAttrib						
AttributeName	itemNo	itemqty	LineItemNum	price	QuantityValue	UnitPrice	UNMATCHED
itemNo	0	0	14	0	0	0	1
itemqty	0	0	1	0	14	0	0
LineItemNum	14	1	0	0	0	0	0
price	0	0	0	0	0	15	0
QuantityValue	0	14	0	0	0	0	1
UnitPrice	0	0	0	15	0	0	0

Figure 3.18 Tableau de contingence des multiples simulations

À l'examen des résultats qui ressortent du tableau de contingence (tableau à deux dimensions[90]), on peut souligner les points suivants :

- Les agents *LineItemNum* et *itemNo* ont pu former un appariement consensuel correct (donnant lieu à la correspondance *LineItemNum*↔*itemNo* attendue par l'utilisateur), avec une fréquence qui s'élève à 14/15. Aussi, on remarque que lors d'une simulation, l'agent *LineItemNum* a formé avec l'agent *itemqty* un appariement consensuel erroné. Quant à l'agent *itemNo*, on remarque qu'il a terminé une simulation sans être capable de former un appariement consensuel (*UNMATCHED*)
- Les agents *UnitPrice* et *price* quant à eux ont fait un parcours sans faute avec une fréquence de 15/15 (*UnitPrice*↔*price*). Ce résultat optionnel revient, sans doute, à l'ambigüité réduite au niveau des noms des agents (i.e. attributs de schémas)
- Les agents *QuantityValue* et *itemqty,* ont eux aussi pu faire un parcours intéressant, formant un appariement consensuel correct (donnant lieu à la correspondance *QuantityValue*↔*itemqty* attendue par l'utilisateur), avec une fréquence qui s'élève à 14/15. Comme pour la première paire d'agents, et à cause du choix erroné des agents *itemqty* et *LineItemNum*, l'agent

[90] De l'anglais « *two-way table* »

QuantityValue a dû terminer une simulation sans être capable de former un appariement consensuel (*UNMATCHED*).

On peut représenter le tableau de contingence[91] ci-dessus (figure 3.18) graphiquement sous la forme d'une carte thermique ou carte de chaleur[92] qui permet une visualisation rapide de l'alignement final (figure 3.19). Cette carte thermique de la matrice des correspondances, représente la fréquence (probabilité/certitude) avec un dégradé de couleurs (couleurs foncées étant une fréquence haute). Cette représentation graphique, grâce à ce dégradé de couleurs, nous permet, non seulement, de visualiser facilement les correspondances, mais encore les erreurs (couleurs pâles) ou les correspondances probables (on peut le voir comme un genre de classification des probabilités *top-k*).

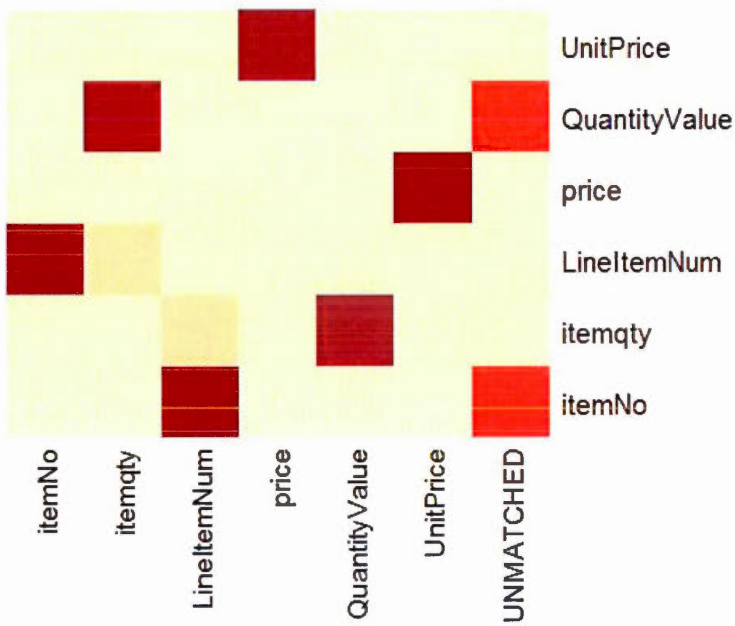


Figure 3.19 Carte thermique du résultat d'une simulation en lot

[91] De l'anglais (*Contingency table*)

[92] De l'anglais (*Heat Map*)

Ci-dessous, un exemple de code *R* permettant de générer une carte thermique à partir du résultat de l'exécution d'une simulation en lot.

```
# Batch run simulation result import
Dataset <- read.table("D:/temp/ModelOutput.2015.Aug.20.19_16_27.txt", header=TRUE, sep=",",
na.strings="NA", dec=".", strip.white=TRUE)
Dataset <- subset(Dataset, subset=tick>1)
.Table <- xtabs(~AttributeName+CandidateCoallitionAttrib, data=Dataset)
# create a matrix
smas_matrix <- data.matrix(.Table)
# import colors lib
library("RColorBrewer")
# display heat map
smas_heatmap <- heatmap(smas_matrix, Rowv=NA, Colv=NA, col = brewer.pal(9,"YlOrRd"),
scale="column", margins=c(15,10))
```

L'analyse statistique avec l'outil R, nous permet de mieux quantifier l'incertitude concernant les appariements trouvés. En effet, on voit bien que, même si durant certaines simulations, on se retrouve avec un appariement erroné, l'analyse statistique de la tendance ou de la fréquence sur un ensemble de simulations, permet de sélectionner des correspondances avec une plus grande confiance.

Figure 3.20 Analyse statistique avec l'outil R

3.4 Collecte et exploitation des données

Cette section traite, à l'aide d'exemples concrets, de l'importance de la collecte et de l'exploitation des données durant les simulations. Ainsi nous présentons deux exemples d'applications concrètes visant l'exploitation des données générées lors des simulations : (i) un réseau des interactions, et (ii) un apparieur prédictif.

3.4.1 Réseau des interactions

Une autre façon d'analyser le résultat des simulations est la représentation de l'évolution des interactions sous la forme d'un *réseau des interactions*.

Analyser le processus d'appariement reviendrait, en quelques sortes, à représenter et à analyser l'histoire des interactions (avec ou sans dimension chronologique). Cette histoire pourrait nous permettre de tirer des enseignements et des conclusions quant à la sélection des combinaisons les plus performantes (le rapport des agents aux autres éléments, notamment les mesures). En effet, on remarque que l'évolution de notre simulation exhibe des aspects de recherche et d'optimisation, des meilleures combinaisons (i.e. d'apparieurs, de fonctions d'agrégation et de paramètres), similaires à ceux que l'on peut observer avec les *algorithmes génétiques*. L'appariement consensuel pourrait être considéré comme la « *fonction d'évaluation* » conduisant cette recherche à la « *algorithmes génétiques* » vers les trajectoires les plus prometteuses.

Dans l'optique de visualiser et d'analyser l'histoire des interactions, on peut par exemple avec l'outil *Gephi* [93] (Bastian *et al.*, 2009) représenter le journal des interactions (généré lors de la simulation par l'outil *Repast Simphny*) en un réseau des interactions (*agent-agent*, *agent-mesure-fonction*). Grâce à ce réseau des interactions, on peut, par exemple, relever visuellement les mesures de similarité les plus performantes (qui ont passé le seuil et participé à un appariement candidat).

---

[93] Gephi version beta 0.8.2

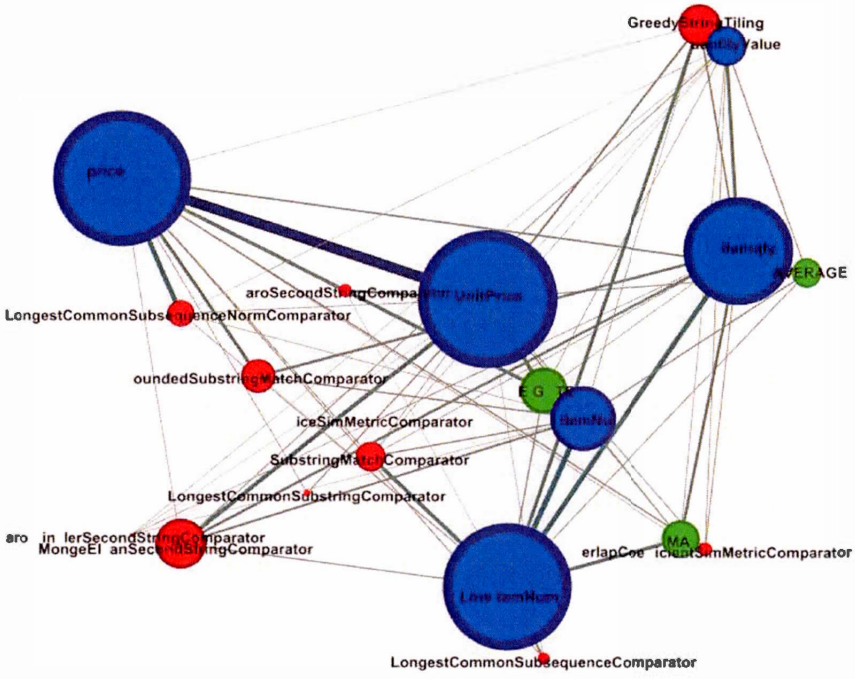


Figure 3.21 Réseau des interactions

Comme le montre la figure 3.21, la force des interactions nous permet de faire une analyse visuelle, non seulement des appariements trouvés mais aussi de la performance des mesures de similarité ainsi que des fonctions d'agrégation.

3.4.2 Apparieur prédictif

Lors de nos expérimentations, on a réalisé que les données générées contenaient beaucoup d'informations intéressantes sur l'exécution de la simulation. Nous nous sommes donc interrogé sur la possibilité d'exploiter ces données pour concevoir un classificateur apprentissage supervisé.

L'idée de base est de générer, pendant les simulations, des exemples étiquetés pour les appariements consensuels (i.e. correct/incorrect) et ce, en comparant les résultats obtenus avec ceux attendus par l'utilisateur. Partant de ce fait, ces exemples étiquetés

serviront ensuite pour la conception d'un nouvel apparieur de type prédictif par le truchement d'une tâche d'apprentissage supervisé. L'apparieur prédictif, grâce à son modèle de la classification, aura donc plus tard, la tâche de prédire pour un agent, à moment donnée de la simulation, le meilleur appariement.

Pour atteindre cet objectif, il a fallu en premier lieu passer par l'étape cruciale de la sélection des attributs d'apprentissage (à partir des données générées) et en deuxième lieu, sélectionner le bon algorithme de classification, en faisant des expérimentations et en évaluant les performances de chaque algorithme, et en dernier lieu, implémenter le modèle de classification obtenu dans la prochaine version prototype (au niveau de l'architecture de l'agent rationnel).

Concernant les attributs d'apprentissage, nous avons ajouté à la logique de l'agent quelques méthodes dont le rôle est le calcul et le stockage des attributs d'apprentissage à des moments différents de la simulation (connaissances mémorisées dans une matrice interne à l'agent et mises à jour, à chaque itération, pendant la phase de perception), constituant au final les instances de l'ensemble d'apprentissage (exemples étiquetés).

Le tableau 3.1 décrit les attributs de la classification sélectionnés pour notre modèle de classification.

Attribut	Description
lexSimTotCalcCount	Le nombre total des calculs de similarité effectués
lexNameSim-AboveThresholdCount	Le nombre total de fois où le score obtenu pour l'attribut *name* dépasse le seuil
lexNameSim-BelowThresholdCount	Le nombre total de fois où le score obtenu pour l'attribut *name* ne dépasse pas le seuil
lexCommentSim-AboveThresholdCount	Le nombre total de fois où le score obtenu pour l'attribut *comment* dépasse le seuil
lexCommentSim-BelowThresholdCount	Le nombre total de fois où le score obtenu pour l'attribut *comment* ne dépasse pas le seuil
lexNameSim-ScoreBestCount	Le nombre total de fois où le score obtenu pour l'attribut *name* a dépassé le meilleur score déjà obtenu
lexCommentSim-ScoreBestCount	Le nombre total de fois où le score obtenu pour l'attribut *comment* a dépassé le meilleur score déjà obtenu
lexNameCommentSim-ConvergenceCount	Le nombre total de fois où il y a eu une convergence entre le meilleur score obtenu pour l'attribut *name* et celui obtenu pour l'attribut *comment*
SemNameCommentSim-Score	Le score de la similarité sémantique (non applicable à cette version du prototype)
SemNameCommentSim-ScoreBest	Valeur booléenne pour la similarité sémantique (non applicable à cette version du prototype)
matchResetCount	Le nombre de fois où l'agent efface ses croyances concernant l'appariement candidat (délai d'attente pour consensus dépassé)
inboundReferralCount	Le nombre de fois où l'agent a été choisi par un autre agent comme appariement candidat
outboundReferralCount	Le nombre de fois où l'agent a choisi un autre agent comme appariement candidat
matchExpert	L'attribut de classification de l'instance courante à savoir appariement correct ou incorrect, au regard des résultats attendus par l'utilisateur : *{false,true}*

Tableau 3.1 Sélection des attributs de la classification

Ensuite, avec l'outil *Weka*[94] (Holmes *et al.*, 1994), à partir de l'ensemble d'apprentissage généré lors de différentes simulations (i.e. différentes simulations pour différents scénarios d'appariement), nous avons conçu un classificateur bayésien, c'est-à-dire basé sur la *règle de Bayes,* dont la formule mathématique est la suivante :

$$P(A|B_1, B_2, \dots, B_n) = \frac{P(B_1, \dots, B_n|A) * P(A)}{P(B_1, \dots, B_n)}$$

(3.6) la formule mathématique de la règle de Bayes

La figure 3.22 montre les attributs de l'ensemble d'apprentissage[95], affichés avec l'interface utilisateur de l'outil *Weka.*

[94] Weka verion 3.7

[95] Sous la forme d'un fichier au format *ARFF*

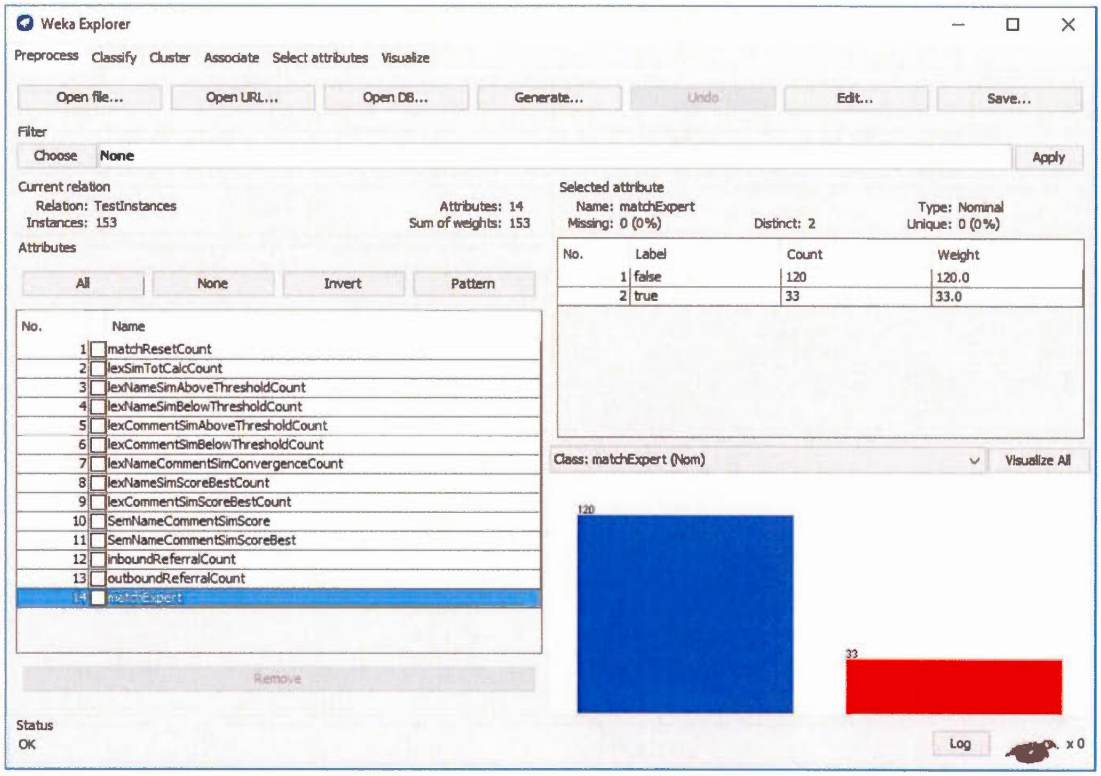


Figure 3.22 Exemples étiquetés avec l'outil *Weka*

L'algorithme de classification qui a été choisi pour la génération du modèle avec *Weka*, est le classificateur bayésien (repose sur une approche probabiliste basée sur la règle de Bayes). La sélection de cet algorithme de classification, s'est basée non seulement après une comparaison théorique des algorithmes populaires, qui peuvent être adéquats pour ce genre de problème de classification, mais encore sur une évaluation de la performance de différents algorithmes de classification (à l'aide de l'outil *Weka Experimenter*).

Notons que ce travail d'expérimentation et d'évaluation des algorithmes de classification n'a pas la prétention d'être exhaustif[96], vu que notre préoccupation

[96] D'un côté parce qu'un travail de « *benchmarking* » plus rigoureux aurait demandé plus de temps sans pour autant avoir une plus-value importante pour notre recherche et d'un autre coté, parce que la portée initiale de notre thèse n'incluait pas la conception d'un nouvel apparieur basé sur l'apprentissage

première était de démontrer (une sorte de preuve de concept) la viabilité de l'exploitation des données de la simulation pour concevoir un apparieur prédictif (exploitant les expériences passées des agents, durant les simulations pour former des appariements consensuels).

Sur le plan théorique, le choix du classificateur bayésien (les réseaux bayésiens) se justifie par sa popularité, sa simplicité et sa performance, comme peut le témoigner plusieurs publications dans la littérature (Pearl, 2011). C'est une approche de classification qui permet, naturellement, de modéliser l'incertitude (probabilité).

3.5 Conclusion

Au terme de ce chapitre, nous pouvons conclure que l'approche de la *simulation multi-agents* ouvre de nouvelles perspectives en appariement automatique de schémas. Les expérimentations préliminaires, présentées et discutées plus-haut, faites sur la première version de notre prototype, en l'occurrence *Reflex-SMAS* (représentant la traduction de notre modèle conceptuel vers un modèle opérationnel basé sur des agents de type réactifs), nous permettent, non seulement de statuer positivement sur la validité de l'idée de voir l'appariement de schémas comme système complexe adaptatif, mais aussi de dire, sans trop courir le risque de se tromper, que le prototype *Reflex-SMAS* a rempli ses promesses.

Les résultats encourageants obtenues avec les agents de type réactifs nous ont poussés à vouloir expérimenter un autre type d'agents, notamment les agents rationnels. Ces derniers auront la particularité de remplacer l'approche naïve pour le choix des actions (faire des choix en se basant sur des règles simplistes), adoptée par les agents réflexifs lors de la phase de décision, par une approche rationnelle leur permettant de raisonner

---

supervisé. Effectivement, cette idée d'exploiter les données, pour la conception d'un apparieur prédictif, s'est imposée d'elle-même dès les premières expérimentations de notre première version de *SMAS* (*Reflex-SMAS*) vu la grande richesse des données générées par un agent, durant la simulation, dans sa quête de la meilleure relation de correspondance chez la population opposée.

et de délibérer sur leurs états ainsi que sur l'état de l'environnement, avant de prendre des décisions.

CHAPITRE IV

VÉRIFICATION ET VALIDATION DU PROTOTYPE REFLEX-SMAS

4.1 Introduction

Dans ce chapitre, nous nous proposons de présenter une validation empirique de notre prototype de simulations multi-agents réflexifs pour l'appariement automatique de schémas (i.e. *Reflex-SMAS*). Nous allons d'abord commencer la présentation de la méthodologie adoptée pour la validation, ensuite, nous enchainons par la description détaillée des résultats obtenus. À la lumière des résultats obtenus, nous discutons à quel point on peut confirmer ou infirmer les hypothèses principales formulées dans cette thèse : (i) la viabilité de l'approche visant à voir l'appariement automatique de schémas comme un système complexe adaptatif, et plus spécifiquement à le modéliser en une simulation multi-agents, (ii) les propriétés intrinsèques de notre modèle conceptuel (qui lui-même hérite les propriétés intrinsèques de la modélisation et de la simulation à base d'agents) comme l'émergence, la stochasticité et l'auto-organisation entre autres, participent à une meilleure gestion de la complexité et de l'incertitude et par voie de conséquence à une augmentation de la qualité du résultat du processus d'appariement de schémas avec une réduction de l'effort de la part de l'utilisateur.

4.2 Objectifs et stratégie de la validation

Maintenant que le modèle conceptuel, le modèle opérationnel et le prototype, pour notre approche de résolution de l'appariement de schémas via des simulations multi-agents, ont été présentés (chapitre 2 et 3), nous arrivons à un jalon important de notre

cheminement de recherche à savoir l'évaluation empirique de notre prototype *Reflex-SMAS*.

Nous cherchons en premier chef, à travers cette évaluation empirique, l'accréditation de l'hypothèse centrale de cette thèse, à savoir la viabilité de proposer une nouvelle approche, issue de la pensée systémique, pour la résolution du problème de l'appariement automatique de schémas. En effet, à travers cette validation nous souhaitons démontrer la viabilité de proposer un changement de paradigme, dans le domaine de l'appariement automatique de schémas, et ce en faisant passer la résolution de la problématique de l'appariement, d'une approche analytique, réductionniste à une approche systémique holistique (basée sur les notions de *complexité* et de *résolution par émergence*).

Ainsi, à l'issue de l'évaluation de notre prototype *Reflex-SMAS* nous souhaitons arriver à démontrer deux principales hypothèses avancées dans thèse, à savoir : d'un côté, (i) la pertinence et la faisabilité de considérer le problème de l'appariement automatique de schémas comme un sujet complexe approprié pour une approche holistique (systémique), permettant de modéliser le système de l'appariement comme un tout, c'est-à-dire, comme un comme ensemble cohérent de composants en interaction, et de l'autre côté, (ii) la possibilité, grâce à un système simple et facile de compréhension, basé sur des règles simples, de réduire l'incertitude, liée principalement à l'ambigüité, et ce en proposant un système d'appariement doté de la faculté d'adaptation qui peut se traduire par des capacités intrinsèques d'auto-configuration (l'adaptation face aux changements) et d'auto-optimisation (la recherche de la meilleure solution) sans aucune intervention manuelle externe.

Plus spécifiquement, nous cherchons à travers cette évaluation de valider les aspects suivants relatifs à notre prototype *Reflex-SMAS* :

- S'il s'agit d'un système d'appariement automatique *efficace* et *efficient*, capable d'une manière autonome en évoluant dans le temps, de s'adapter, de s'auto-organiser et ainsi de faire émerger la solution pour n'importe quel scénario d'appariement :
    - Une efficacité qui doit se traduire principalement, de par une augmentation de la qualité de l'alignement trouvé et une meilleure quantification de l'incertitude (réduction de l'incertitude).
    - L'efficience, quant à elle, doit se matérialiser par la réduction de l'effort nécessaire à cette efficacité notamment la réduction de l'effort utilisateur nécessaire à la configuration ou à l'optimisation du système d'appariement et cela sans nuire considérablement au temps de réponse qui devrait rester dans les limites du viable et de l'acceptable (surtout pour l'utilisation de l'appariement dans des environnements hautement dynamiques).
- S'il s'agit vraiment d'un système de compréhension facile, et par conséquence pourrait afficher un haut degré de maintenabilité (e.g. ajout de nouveaux apparieurs).

D'ores et déjà, nous savons, grâce aux expérimentations préliminaires (chapitre 3), que notre prototype est dans une large mesure conforme au point cités plus-haut ; il nous reste maintenant à en apporter la preuve par une expérimentation plus poussée. Pour ce faire nous devons d'abord établir la stratégie de la preuve, c'est-à-dire la façon dont nous allons nous y prendre pour vérifier nos hypothèses de recherche.

En ce qui nous concerne, la stratégie de la preuve est centrée autour de la conduite d'expérimentations, de la collecte et de l'analyse de données issues de ces expérimentations.

Ainsi, l'approche de validation que nous avons adoptée se veut être une hybridation de deux approches de validation issues de deux domaines différents, notamment le

domaine de l'appariement automatique de schémas et celui de la modélisation et de la simulation multi-agents. Il s'agit d'une part, d'un mode d'évaluation populaire[97], dans le domaine de l'appariement automatique de schémas, à savoir la comparaison des résultats obtenus avec ceux attendus par l'utilisateur (Bellahsene Bonifati Duchateau *et al.*, 2011) , et d'autre part, d'un type de validation préconisé pour les modèles de simulation multi-agents à savoir la *validation empirique* (Remondino et Correndo, 2006), basée sur la comparaison des résultats obtenues du modèle par rapport à ce que l'on peut observer sur le vrai système.

La stratégie mise en place pour la validation, de notre modèle de simulation multi-agents, mais aussi des résultats pour la validation de nos hypothèses, consiste en :

- la définition des scénarios d'appariement différents, de telle sorte nous puissions tester le prototype dans différentes situations;
- l'évaluation de la performance en comparant les résultats obtenus, par le prototype *Reflex-SMAS*, avec les résultats attendus par l'utilisateur pour les trois scénarios d'appariement.

Nous avons fait le choix de ne pas procéder à une expérimentation basée sur la *comparaison des résultats*[98] de *Reflex-SMAS* avec ceux obtenus par d'autres outils existants. Le but premier de notre démarche est de confirmer ou d'infirmer la viabilité de ce changement de paradigme l'appariement automatique de schémas (i.e. la vision de l'appariement de schémas comme système complexe adaptatif et sa modélisation

---

[97] « *Measuring the effectiveness of a mapping or matching tool means measuring whether (or how much) the tool can fulfill its expectations for a given task. In the case of matching, an expert user typically knows what the correct matches are, and the matching tool is expected to find them. Thus, evaluating its effectiveness boils down to a comparison between the expected set of matchings and the set of matchings that the tool generated.* » (Bellahsene Bonifati Duchateau *et al.*, 2011)

[98] *benchmarking*

comme une simulation multi-agents) qui au meilleur de nos connaissances n'a jamais été tenté dans le domaine de de l'appariement de schémas.

4.3 Expérimentations

4.3.1 Méthodologie

Les deux grands axes de notre méthodologie de validation peuvent être déclinées comme suit : (i) la définition des scénarios d'appariement, ensuite (ii) la conduite des expérimentations et la compilation des résultats.

En ce qui a trait à la définition des scénarios, ils ont été développés en prenant en considération les critères suivants :

- des scénarios de différentes tailles (i.e. en fonction du nombre d'éléments dans les schémas);
- des scénarios exhibant différents niveaux d'hétérogénéité lexicale (i.e. au niveau du nom et de la description des éléments de schémas);
- des scénarios touchant à différents domaines d'affaire.

Nous avons donc défini 3 scénarios d'appariement, en l'occurrence les scénarios « *Person* », « *Order* » et « *Travel* », de tailles différentes, de niveaux d'hétérogénéité lexicale différents et représentant des domaines d'affaire différents.

Le premier scénario « *Person* » est un scénario de petite taille (i.e. 6 éléments de schémas) et dont le niveau d'hétérogénéité lexicale pourrait être considéré comme moyen. Ce scénario a été inspiré d'un exemple, utilisé pour l'évaluation de l'outil *YAM* (Duchateau et Bellahsene, 2014), que nous avons ensuite adapté en ajoutant des commentaires au niveau de tous les éléments de schémas.

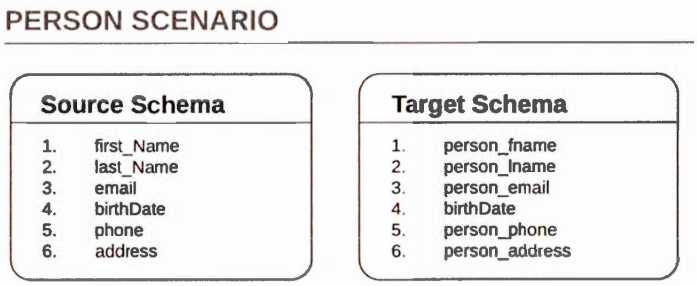


Figure 4.1 Scénario d'appariement « *Person* »

Le second scénario « *Order* » est un scénario de taille moyenne (i.e. 8 éléments de schémas) exhibant un niveau d'hétérogénéité lexicale élevé. Ce scénario a été inspiré d'un exemple utilisé pour l'évaluation de l'outil *YAM* (Duchateau et Bellahsene, 2014) que nous avons modifié dans le but de l'étoffer avec l'ajout de nouveaux éléments de schémas (pour faire augmenter la taille du scénario), ainsi qu'avec l'ajout des commentaires au niveau de tous les éléments de schémas.

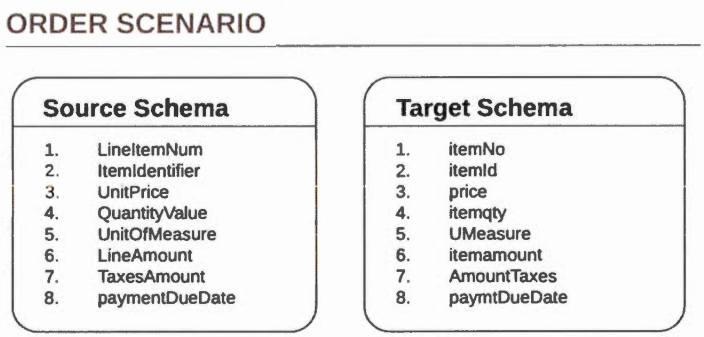


Figure 4.2 Scénario d'appariement « *Order* »

Le dernier scénario « *Travel* » est un scénario qui affiche une hétérogénéité lexicale relativement basse (i.e. une hétérogénéité lexicale allant de basse à moyenne) ainsi qu'une taille qui peut être considérée comme une grande taille (i.e. 15 éléments par schéma) si nous la comparons aux deux autres scénarios. Ce scénario a aussi été inspiré

des exemples utilisés pour l'évaluation de l'outil *YAM* (Duchateau et Bellahsene, 2014) (complété avec des éléments provenant de diverses schémas concernant le voyage disponible sur le site de *OpenTravel* [99]).

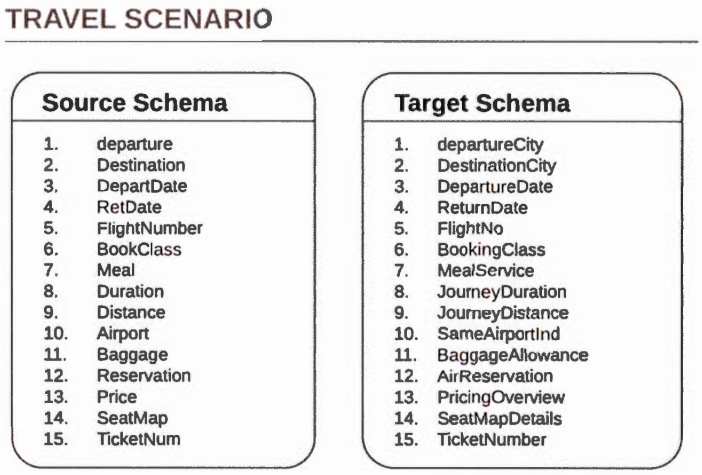


Figure 4.3 Scénario d'appariement « *Travel* »

Dans le but d'évaluer la pertinence ainsi que le niveau de difficulté que peut représenter les scénarios d'appariement « *Person* », « *Order* » et « *Travel* », nous avons décidé de les soumettre à des tests, à l'aide de l'outil *COMA*[100].

Les résultats obtenus avec l'outil COMA, tel qu'illustré par la figure 4.4, montrent clairement que les scénarios « *Person* », « *Order* » et « *Travel* », de par leur niveau d'hétérogénéité, devraient poser des défis intéressants quant à la résolution de toutes les correspondances attendues.

[99] L'url pour accéder aux schémas : http://www.opentravel.org/Specifications/PastSpecs.aspx

[100] Les tests ont exploité les stratégies (i.e. flux de travail) par défaut (i.e. *$NodesNameS* et *$NodesS*)

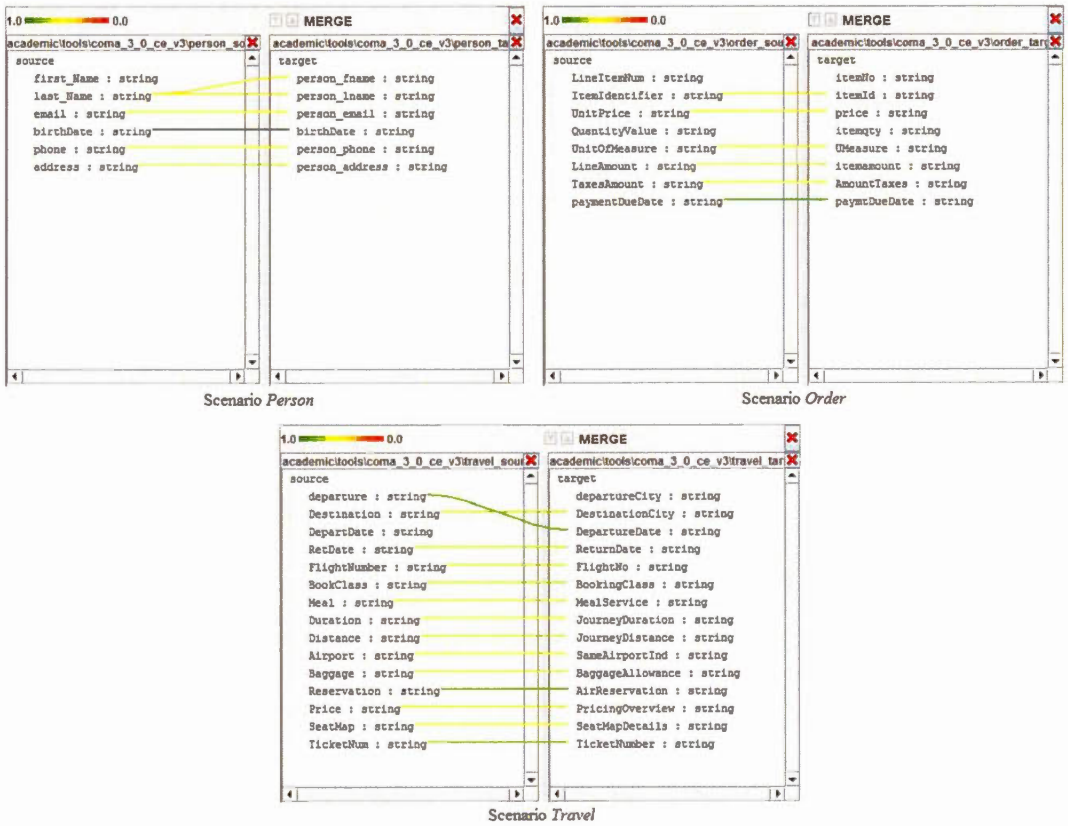


Figure 4.4 Résultats de l'évaluation des scénarios avec l'outil COMA

Concernant la conduite des expérimentations et la compilation des résultats, nous avons décidé d'exécuter une série de 3 méta-simulations (chaque méta-simulation comporte 10 simulations) pour chaque scénario. Le résultat final pour l'appariement du scénario (i.e. alignement) sera basé sur le résultat de chaque méta-simulation, c'est-à-dire que l'analyse statistique faite sur chaque méta-simulation (calcul de la fréquence de l'apparition d'une correspondance sur les 10 simulations de la méta-simulation) détermine la solution de l'alignement trouvé pour le scénario d'appariement (i.e. les correspondances). L'exécution de 3 méta-simulations, pour chaque scénario, a pour objectif de vérifier la répétabilité des résultats.

Après l'exécution de chaque méta-simulation, les données concernant l'exécution ainsi que les résultats obtenus sont compilés à deux niveaux :

- au niveau de la simulation, on capture des données concernant l'exécution de chaque simulation (e.g. nombre d'itération). Ensuite, nous calculons la performance de chaque simulation (en utilisant des mesures d'évaluation de la performance à savoir la *précision*, le *rappel* ou *F-mesure*), et ce en comparant les correspondances obtenues par la simulation aux correspondances attendues;
- au niveau de la méta-simulation, nous obtenons le résultat final pour l'appariement du scénario, à l'aide de l'analyse statistique de la méta-simulation qui se traduit par un calcul de la fréquence de l'apparition d'une correspondance sur les 10 simulations qui la composent. À la suite de quoi, nous calculons la performance de chaque méta-simulation (en utilisant des mesures d'évaluation de la performance à savoir la *précision*, le *rappel* ou F-mesure), et ce en comparant les correspondances obtenues par la méta-simulation aux correspondances attendues.

L'évaluation de la qualité de l'alignement trouvé (i.e. l'ensemble des correspondances trouvées) par les outils de l'appariement automatique de schémas, passe souvent par l'utilisation de certaines mesures de performance, à savoir la *précision*, le *rappel* et la *F-mesure* (Bellahsene Bonifati Duchateau *et al.*, 2011). Ils permettent de mesurer la performance par rapport à l'alignement attendu. Ces mesures sont aussi populaires dans le domaine de l'extraction de l'information.

Le tableau des contingences suivant est à la base du calcul de ces mesures de performance.

	Correspondances correctes	***Correspondances incorrectes***
Correspondances correctes trouvées	VP (vrais positifs)	FP (faux positifs)
Correspondances incorrectes trouvées	FN (faux négatifs)	VN (vrais négatifs)

Tableau 4.1 Tableau des contingences pour les mesures de performance

Dans le contexte de l'appariement automatique de schémas, les valeurs du tableau de contingences peuvent êtres décrits de la manière suivante :

- *VP* (vrais positifs) : Toutes les correspondances correctes identifiées comme correctes par l'outil
- *FP* (faux positifs) : Toutes les correspondances incorrectes identifiées comme correctes par l'outil
- *FN* (faux négatifs) : Toutes les correspondances correctes identifiées comme incorrectes par l'outil (incluant les correspondances non trouvées)
- *VN* (faux négatifs) : Toutes les correspondances incorrectes identifiées comme incorrectes par l'outil

La première mesure pour l'évaluation de la performance est la mesure *Précision*[101]. Cette mesure calcule la proportion des correspondances correctes trouvées par rapport à l'ensemble des correspondances trouvées par l'outil (i.e. la somme des correspondances correctes trouvées et des correspondances incorrectes trouvées) :

[101] De l'anglais « *Precision* »

$$Précision = \frac{VP}{VP + FP}$$

(4.1) Évaluation de la performance : *Précision*

La seconde mesure est la mesure *Rappel*[102] qui calcule la proportion des correspondances correctes trouvées par l'outil par rapport à toutes les correspondances pertinentes attendues par l'utilisateur (i.e. la somme des correspondances correctes trouvées et des correspondances correctes non trouvées par l'outil) :

$$Rappel = \frac{VP}{VP + FN}$$

(4.2) Évaluation de la performance : *Rappel*

Et enfin, la dernière mesure est la mesure *F-mesure*[103] qui se trouve être une mesure qui combine la *précision* et le *rappel* en calculant leur moyenne harmonique :

$$F - mesure = 2 \times \frac{(Précision \times Rappel)}{(Précision + Rappel)}$$

(4.3) Évaluation de la performance : *F-mesure*

4.3.2 Configuration

Pour ce qui est de la configuration matérielle que nous avons utilisée pour conduire les expérimentations, il s'agit d'un ordinateur portable disposant d'un micro-processeur multi-cœur (i.e. 4 cœurs) avec 32 GO de mémoire.

---

[102] De l'anglais « Recall »

[103] De l'anglais « *F-Measure* »

Type composant	*Spécifications*	*Description*
Micro-processeur	Intel(R) Core(TM) i7-4810MQ CPU @ 2.80GHz	Nb. de cœurs = 4 Nb. de threads = 8 Fréquence de base = 2.8 GHz Fréquence Turbo maxi = 3.8 GHz
Mémoire	32.00 Go	

Tableau 4.2 Spécification matérielle

Dans l'optique de mettre en relief la capacité de notre prototype de tirer profit du parallélisme, nous avons décidé, vu que la configuration matérielle le permet, de faire exécuter chaque simulation (composant la méta-simulation) dans une instance séparée. Ce qui veut dire pour un scénario d'appariement donné, que pour chaque méta-simulation, l'outil *Repast Simphony*, démarre et fait exécuter en parallèle 10 instances (i.e. 10 machine virtuelle Java avec une taille de 1 Go[104]).

```
<?xml version="1.0" ?>
<sweep runs="10">
<parameter name="randomSeed" type="constant" constant_type="int" value="10"></parameter>
<parameter name="runLength" type="constant" constant_type="double" value="10000"></parameter>
</sweep>
```

À la fin de la méta-simulation, l'outil *Repast Simphony,* procède à une consolidation du résultat de chaque simulation (i.e. chaque machine virtuelle Java) dans un seul et même fichier pour la méta-simulation.

[104] *VM Arguments = -Xmx1024M*

4.3.3 Résultats

Ci-dessous, la synthèse des résultats obtenus à la suite des expérimentations conduites pour l'évaluation de l'outil *ReflexSMAS.*

4.3.3.1 Scénario « *Person* »

En examinant les résultats de l'analyse statistique de chaque méta-simulation nous réalisons que pour ce scénario notre prototype a été capable de trouver correctement toutes les correspondances attendues (100% des correspondances).

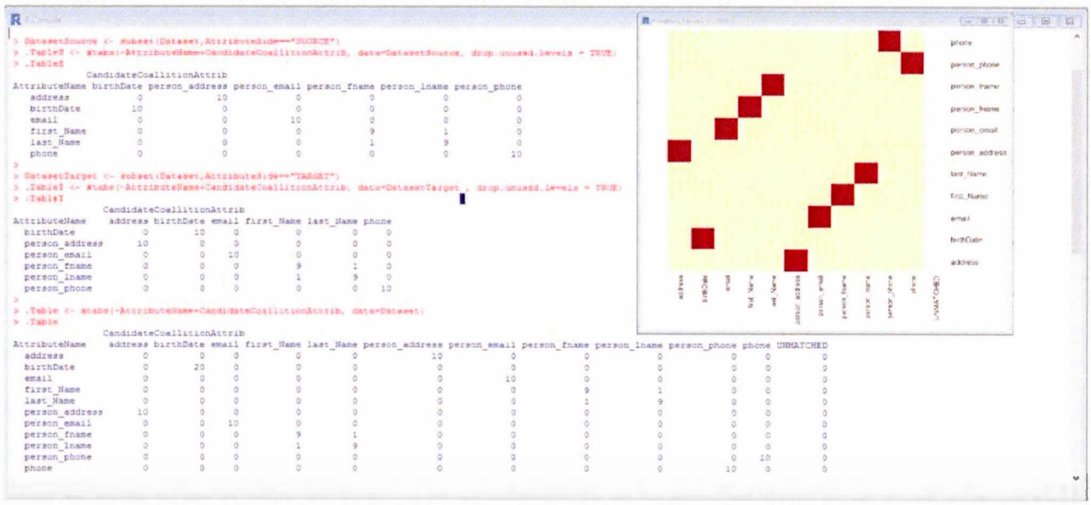


Figure 4.5 Résultat de la méta-simulation n°1 pour le scénario « *Person* »

Les deux premiers tableaux de la figure 4.5 montrent le résultat de la méta-simulation n° 1 pour le scénario « *Person* ». Le premier tableau montre la perspective des agents représentant le schéma *source*, tandis que le deuxième tableau montre les résultats sous la perspective des agents représentant le schéma *destination.* Le dernier tableau montre une vue globale des deux perspectives. Ce dernier est représenté graphiquement sous la forme d'une carte de chaleur.

Examinons maintenant les résultats détaillés pour le scénario « *Person* ». Les résultats, des 3 différentes méta-simulations sont compilés dans le tableau ci-dessous.

	Itérations	Nbr. Corresp. à trouver	Nbr. Corresp. trouvées correctes	Nbr. Corresp. trouvées incorrectes	Nbr. Corresp. non trouvées	Précision	Rappel	F-mesure
Méta-sim. 1	**629.4**	**12**	**11.8**	**0.2**	**0**	**98%**	**98%**	**98%**
1	689	12	12	0	0	100%	100%	100%
2	2151	12	10	2	0	83%	83%	83%
3	572	12	12	0	0	100%	100%	100%
4	542	12	12	0	0	100%	100%	100%
5	678	12	12	0	0	100%	100%	100%
6	669	12	12	0	0	100%	100%	100%
7	238	12	12	0	0	100%	100%	100%
8	549	12	12	0	0	100%	100%	100%
9	190	12	12	0	0	100%	100%	100%
10	16	12	12	0	0	100%	100%	100%
Méta-sim. 2	**492.3**	**12**	**12**	**0**	**0**	**100%**	**100%**	**100%**
1	203	12	12	0	0	100%	100%	100%
2	186	12	12	0	0	100%	100%	100%
3	306	12	12	0	0	100%	100%	100%
4	240	12	12	0	0	100%	100%	100%
5	334	12	12	0	0	100%	100%	100%
6	1322	12	12	0	0	100%	100%	100%
7	750	12	12	0	0	100%	100%	100%
8	15	12	12	0	0	100%	100%	100%
9	1333	12	12	0	0	100%	100%	100%
10	234	12	12	0	0	100%	100%	100%
Méta-sim. 3	**999.4**	**12**	**11.8**	**0.2**	**0**	**98%**	**98%**	**98%**
1	334	12	12	0	0	100%	100%	100%
2	172	12	12	0	0	100%	100%	100%
3	931	12	12	0	0	100%	100%	100%
4	498	12	12	0	0	100%	100%	100%
5	4117	12	10	2	0	83%	83%	83%
6	185	12	12	0	0	100%	100%	100%
7	5	12	12	0	0	100%	100%	100%
8	9	12	12	0	0	100%	100%	100%
9	651	12	12	0	0	100%	100%	100%
10	3092	12	12	0	0	100%	100%	100%

Tableau 4.3 Mesure des performances pour le scénario « *Person* »

D'entrée de jeu, nous souhaitons clarifier un point. En examinant les données du tableau 4.3, on constate tout d'abord que le nombre *Nbr. Corresp. à trouver* est égal à la somme des deux éléments de schémas c'est-à-dire 12 alors que pour nous, le nombre des correspondances à trouver doit être égal à 6 paires d'éléments de schémas. Cela est dû au fait que notre modèle, de par la nature autonome et décentralisé de ses agents (absence de matrice de similarité), offre la perspective des agents des deux groupes quant aux correspondances trouvées, ce qui fait que le nombre des correspondances à trouver est présenté selon le point de vue des agents de chaque groupe.

Ceci nous amène à une autre remarque, concernant les chiffres au niveau des méta-simulations. Il s'agit d'une agrégation (i.e. moyenne) des données observées pour les différentes simulations individuelles (i.e. les 10 simulations constituant la méta-simulation).

En analysant les données du tableau 4.3, on constate, que pratiquement lors de toutes les simulations individuelles, notre prototype a été capable de trouver les bonnes correspondances (i.e. *Nbr. Corresp. trouvées correctes = 12*), ce qui fait que les mesures de performance (i.e. *Précision, Rappel, F-mesure*), que ce soit au niveau de chaque simulation ou au niveau de la méta-simulation (i.e. moyenne), sont très élevées.

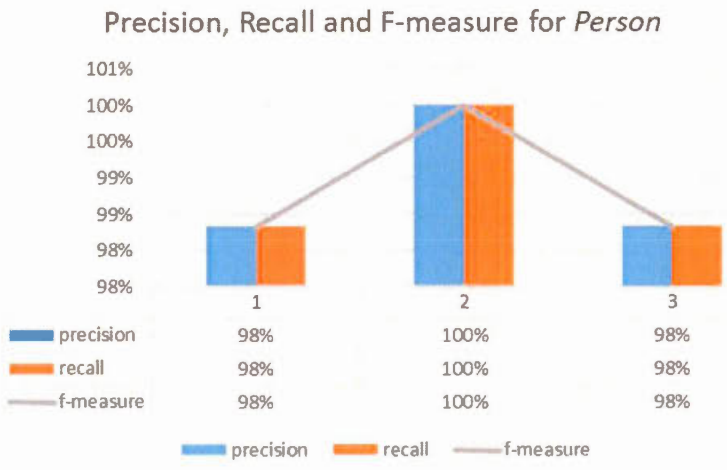


Figure 4.6 Précision, Rappel et F-mesure pour le scénario « *Person* »

Nous pouvons donc conclure que, pour un tel scénario (schéma de petite taille ayant une hétérogénéité lexicale moyenne), le prototype est capable de trouver, grâce à l'analyse statistique sur une méta-simulation, toutes les correspondances attendues.

4.3.3.2 Scénario « *Order* »

L'analyse statistique sur les expérimentations concernant le scénario d'appariement « *Order* », dont la taille est considérée comme moyenne (i.e. 8 éléments), et dont l'hétérogénéité lexicale est considérée comme élevée, nous a permis de constater que notre prototype performe encore une fois très bien dans la mesure où il a été capable de trouver correctement, pour chaque méta-simulation, toutes les correspondances attendues (100% des correspondances).

Pour expliquer comment notre prototype peut arriver à des résultats aussi bons (100% des correspondances correctement trouvées) prenons, par exemple la correspondance attendue *QuantityValue↔itemqty*. En jetant un coup d'œil rapide sur la carte de chaleur de la figure 4.7, illustrant l'analyse statistique de la méta-simulation n°2 pour le scénario « *Order* », on constate que malgré l'échec de certaines simulations à trouver

la correspondance *QuantityValue↔itemqty* (les éléments *QuantityValue* et *itemqty* finissent avec le statut *UNMATCHED*), il n'en demeure pas moins, qu'en prenant en compte l'ensemble des résultats pour les simulations individuelles, la correspondance *QuantityValue↔itemqty* fait bel et bien partie du résultat final. Compte tenu du fait que certaines simulations individuelles arrivent à trouver la correspondance attendue et aussi compte tenu du fait que c'est la fréquence d'une correspondance au niveau de la méta-simulation qui détermine le résultat final par conséquent, l'échec d'une ou plusieurs simulations ne conduit pas nécessairement à l'échec de trouver la bonne correspondance.

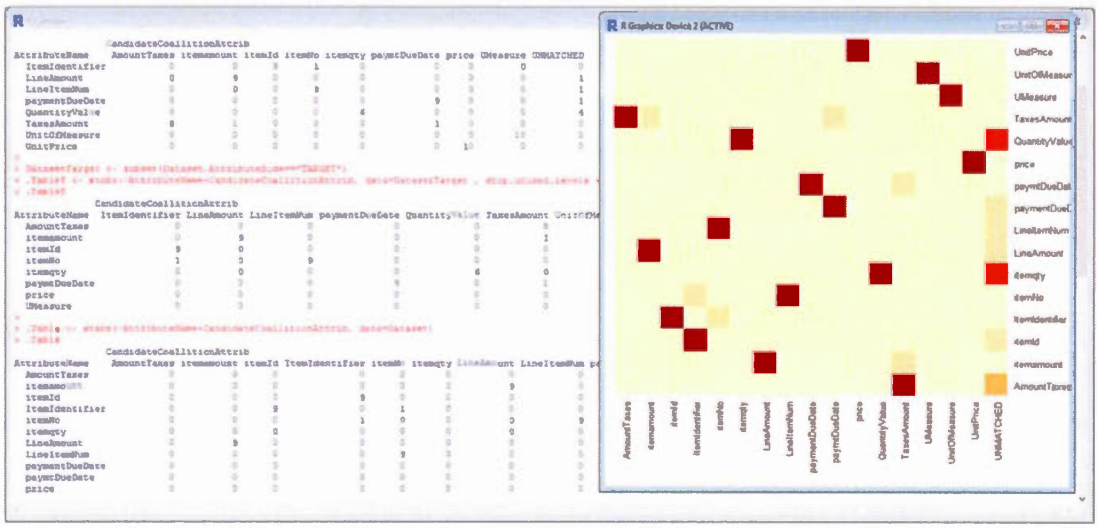


Figure 4.7 Résultat de la méta-simulation n°2 pour le scénario « *Order* »

À l'examen des résultats compilés dans le tableau 4.4 nous réalisons que nous avons obtenu, pour ce scénario, des valeurs assez élevées pour les mesures de performances (e.g. *F-Mesure* supérieure au niveau des méta-simulations à 84%). En revanche nous remarquons une nette augmentation de la moyenne du nombre d'itérations pour les simulation (qui représente en quelques sortes l'effort déployé par les agents pour former des appariements consensuels), par rapport au premier scénario. Nous remarquons, par exemple, pour la méta-simulation n° 3, qu'une grande majorité de

simulations ont atteint le nombre maximal des itérations avant de trouver la solution complète.

	Itérations	Nbr. Corresp. à trouver	Nbr. Corresp. trouvées correctes	Nbr. Corresp. trouvées incorrectes	Nbr. Corresp. non trouvées	Précision	Rappel	F-mesure
Méta-sim. 1	**8196.4**	**16**	**13**	**1.8**	**1.2**	**87%**	**81%**	**84%**
1	10000	16	14	2	0	88%	88%	88%
2	10000	16	10	2	4	83%	63%	71%
3	6826	16	16	0	0	100%	100%	100%
4	10000	16	10	4	2	71%	63%	67%
5	10000	16	14	0	2	100%	88%	93%
6	9502	16	12	4	0	75%	75%	75%
7	10000	16	14	0	2	100%	88%	93%
8	10000	16	8	6	2	57%	50%	53%
9	2001	16	16	0	0	100%	100%	100%
10	3635	16	16	0	0	100%	100%	100%
Méta-sim. 2	**8044.1**	**16**	**14**	**0.6**	**1.4**	**95%**	**88%**	**91%**
1	9636	16	16	0	0	100%	100%	100%
2	10000	16	14	0	2	100%	88%	93%
3	10000	16	12	2	2	86%	75%	80%
4	6673	16	16	0	0	100%	100%	100%
5	10000	16	14	0	2	100%	88%	93%
6	10000	16	14	0	2	100%	88%	93%
7	5706	16	16	0	0	100%	100%	100%
8	10000	16	6	4	6	60%	38%	46%
9	6203	16	16	0	0	100%	100%	100%
10	2223	16	16	0	0	100%	100%	100%
Méta-sim. 3	**9119.2**	**16**	**13.6**	**0.4**	**2**	**97%**	**85%**	**90%**
1	8609	16	16	0	0	100%	100%	100%
2	10000	16	14	0	2	100%	88%	93%
3	10000	16	14	0	2	100%	88%	93%
4	10000	16	14	0	2	100%	88%	93%
5	10000	16	14	0	2	100%	88%	93%
6	10000	16	10	2	4	83%	63%	71%
7	10000	16	10	2	4	83%	63%	71%
8	10000	16	14	0	2	100%	88%	93%
9	2583	16	16	0	0	100%	100%	100%
10	10000	16	14	0	2	100%	88%	93%

Tableau 4.4 Mesure des performances pour le scénario « *Order* »

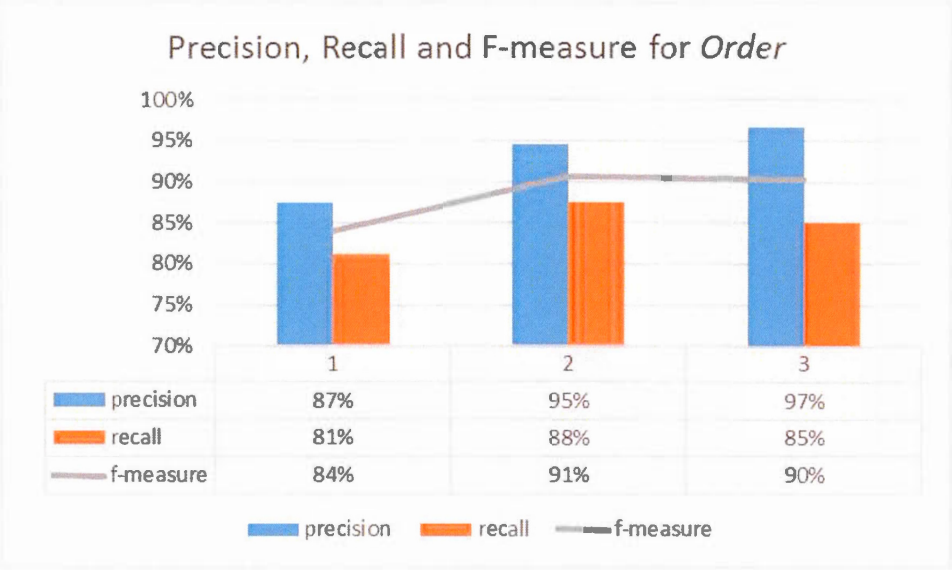


Figure 4.8 Précision, Rappel et F-mesure pour le scénario « *Order* »

Nous pouvons donc conclure que, pour un tel scénario (schéma de taille moyenne ayant une hétérogénéité lexicale élevée), le prototype a été capable de trouver, pour chaque méta-simulation, toutes les correspondances attendues. En revanche, on peut dire que l'hétérogénéité lexicale élevée corrèle négativement le nombre d'itérations nécessaires pour trouver la solution au niveau de chaque simulation.

4.3.3.3 Scénario « *Travel* »

Examinons maintenant le scénario d'appariement « *Travel* » qui exhibe une hétérogénéité lexicale relativement basse (i.e. une hétérogénéité lexicale de basse à moyenne), ainsi qu'une taille qui peut être considérée grande (i.e. 15 éléments par schéma).

D'emblée et à la lumière des résultats de l'analyse statistique des méta-simulations (la carte thermique ou encore les tableaux de contingence), nous pouvons dire encore une fois, que toutes les correspondances attendues ont été trouvées correctement (100%).

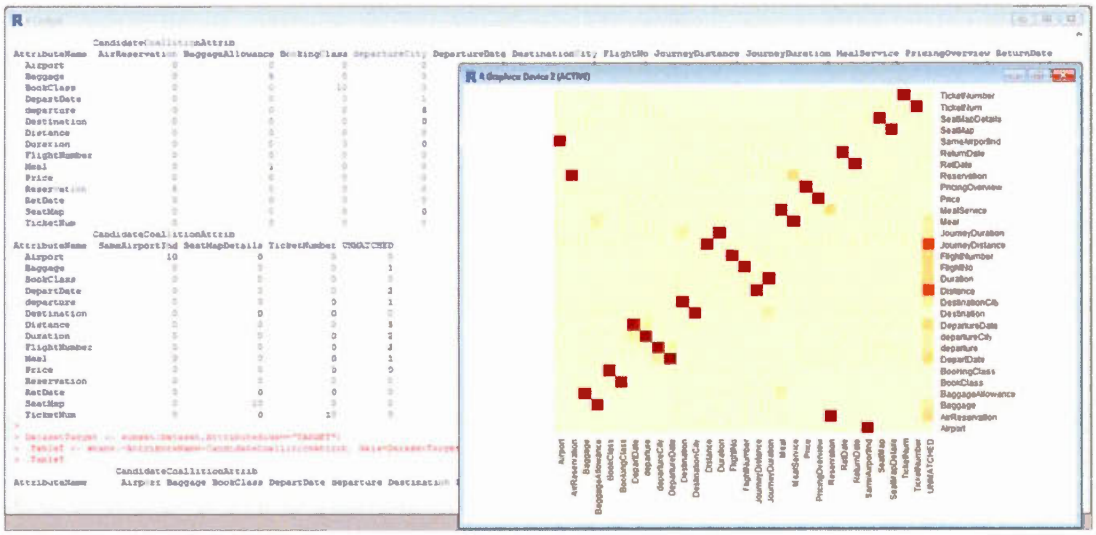


Figure 4.9 Résultat de la méta-simulation n°3 pour le scénario « *Travel* »

Les résultats, des 3 différentes méta-simulations (incluant le détail de chaque simulation), pour le scénario « *Travel* », sont compilés dans le tableau 4.5. En examinant les résultats, nous remarquons qu'avec une grande taille pour les schémas et une hétérogénéité lexicale relativement basse, le prototype a été capable de trouver souvent toutes les bonnes correspondances avec très peu de correspondances incorrectes signalées. Les résultats concernant les correspondances non trouvées (i.e. *UNMATCHED*) peuvent être expliqués par la durée des simulations, fixée à un nombre maximal d'itérations de 10.000 itérations. Donc malheureusement, la durée maximale a été un frein pour trouver toutes les correspondances attendues lors de certaines simulations. Les mesures de l'évaluation peuvent être améliorées si nous augmentons le nombre maximal d'itérations de 10.000 à 15.000. Cependant dans un souci de consistance, nous avons décidé de garder le même nombre maximal.

	itérations	Nbr. Corresp. à trouver	Nbr. Corresp. trouvées correctes	Nbr. Corresp. trouvées incorrectes	Nbr. Corresp. non trouvées	Précision	Rappel	F-mesure
Méta-sim. 1	**8462.3**	**30**	**28.6**	**0**	**1.4**	**100%**	**95%**	**98%**
1	5384	30	30	0	0	100%	100%	100%
2	10000	30	28	0	2	100%	93%	97%
3	10000	30	28	0	2	100%	93%	97%
4	10000	30	26	0	4	100%	87%	93%
5	10000	30	28	0	2	100%	93%	97%
6	6442	30	30	0	0	100%	100%	100%
7	6473	30	30	0	0	100%	100%	100%
8	9234	30	30	0	0	100%	100%	100%
9	10000	30	26	0	4	100%	87%	93%
10	7090	30	30	0	0	100%	100%	100%
Méta-sim. 2	**9904.4**	**30**	**28**	**0**	**2**	**100%**	**93%**	**97%**
1	10000	30	28	0	2	100%	93%	97%
2	9723	30	28	0	2	100%	93%	97%
3	10000	30	28	0	2	100%	93%	97%
4	10000	30	28	0	2	100%	93%	97%
5	9992	30	30	0	0	100%	100%	100%
6	10000	30	28	0	2	100%	93%	97%
7	10000	30	26	0	4	100%	87%	93%
8	9329	30	30	0	0	100%	100%	100%
9	10000	30	26	0	4	100%	87%	93%
10	10000	30	28	0	2	100%	93%	97%
Méta-sim. 3	**9479.5**	**30**	**26.6**	**0.8**	**2.6**	**97%**	**89%**	**92%**
1	6549	30	30	0	0	100%	100%	100%
2	10000	30	26	0	4	100%	87%	93%
3	10000	30	20	4	6	83%	67%	74%
4	10000	30	24	4	2	86%	80%	83%
5	8590	30	30	0	0	100%	100%	100%
6	10000	30	26	0	4	100%	87%	93%
7	10000	30	26	0	4	100%	87%	93%
8	10000	30	28	0	2	100%	93%	97%
9	9656	30	30	0	0	100%	100%	100%
10	10000	30	26	0	4	100%	87%	93%

Tableau 4.5 Mesure des performances pour le scénario « *Travel* »

On peut remarquer que la moyenne de la mesure *F-mesure* obtenue pour chaque simulation de la méta-simulation est supérieure à 92% ce qui représente un excellent taux.

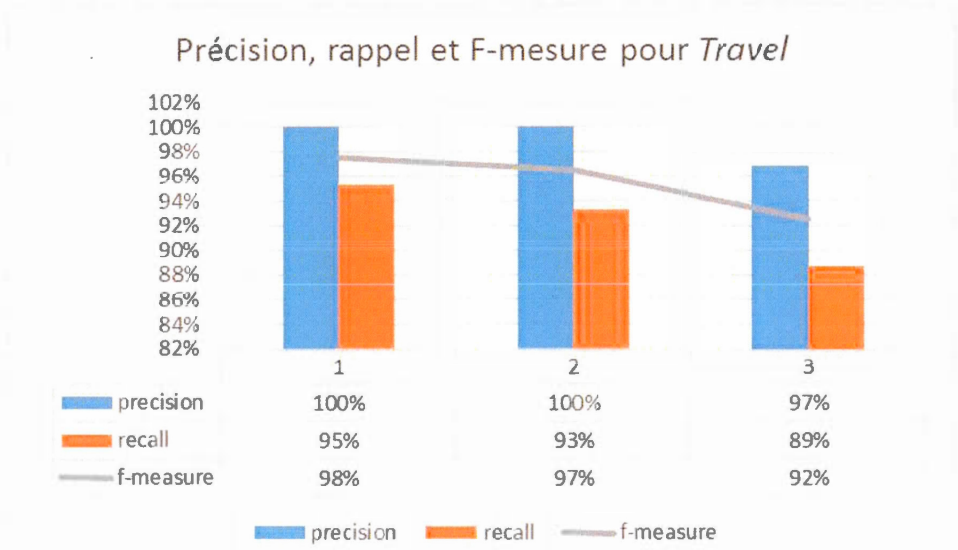


Figure 4.10 *Précision*, *Rappel* et *F-mesure* pour le scénario « *Travel* »

Nous pouvons donc conclure que, pour un tel scénario (schéma de grande taille ayant une hétérogénéité lexicale relativement basse), le prototype a été capable de trouver, pour chaque méta-simulation, toutes les correspondances attendues. En revanche, on peut dire que la grande taille des schémas corrèle négativement le nombre d'itérations nécessaires pour trouver la solution au niveau de chaque simulation.

4.4 Synthèse et discussion des résultats

À la lumière des résultats obtenus, nous discutons dans cette section, la performance de notre prototype au regard des deux aspects suivants :

- Efficacité : cet aspect concerne l'évaluation de la qualité de l'alignement trouvé pour chaque méta-simulation et pour chaque scénario. La réduction de l'incertitude sur l'alignement trouvé, pour notre approche, passe par une

meilleure quantification de l'incertitude (analyse statistique sur des méta-simulations).

- Efficience : cet aspect concerne l'évaluation de l'effort nécessaire à cette efficacité notamment la réduction de l'effort utilisateur nécessaire à la configuration ou à l'optimisation du système d'appariement. Par ailleurs, il concerne aussi l'évaluation du temps de réponse nécessaire à l'obtention du résultat final (i.e. l'alignement) pour un scénario d'appariement donné.

Concernant l'aspect efficacité, la stratégie de preuve repose principalement sur l'évaluation de la performance obtenue par notre prototype pour les différents scénarios utilisés. Ces derniers, comme nous l'avons déjà expliqué, affichent un niveau d'hétérogénéité et d'ambiguïté lexicale allant de bas à élevé (voir même très élevé pour certains éléments de schémas).

Le tableau ci-dessous, présente les résultats combinés pour tous les scénarios. Rappelons que les mesure *précision*, *rappel* et *F-mesure* sont une moyenne des simulations de chaque méta-simulation.

Scénario	Méta sim.no	Itérations	Précision	Rappel	F-mesure
Person-1	1	629.4	98%	98%	98%
Person-2	2	492.3	100%	100%	100%
Person-3	3	999.4	98%	98%	98%
Order-1	1	8196.4	87%	81%	84%
Order-2	2	8044.1	95%	88%	91%
Order-3	3	9119.2	97%	85%	90%
Travel-1	1	8462.3	100%	95%	98%
Travel-2	2	9904.4	100%	93%	97%
Travel-3	3	9479.5	97%	89%	92%

Tableau 4.6 Résultats combinés pour tous les scénarios

Ainsi que le met en évidence la figure 4.11, la performance de notre prototype (la moyenne des simulations d'une méta-simulation) est très élevée pour tous les scénarios

(avoisinant le 100% pour les scénarios « *Person* » et « *Travel* »). Toutefois, nous pouvons remarquer une légère baisse pour le scénario « *Order* » à cause sans doute du fait qu'il exhibe une hétérogénéité lexicale élevée.

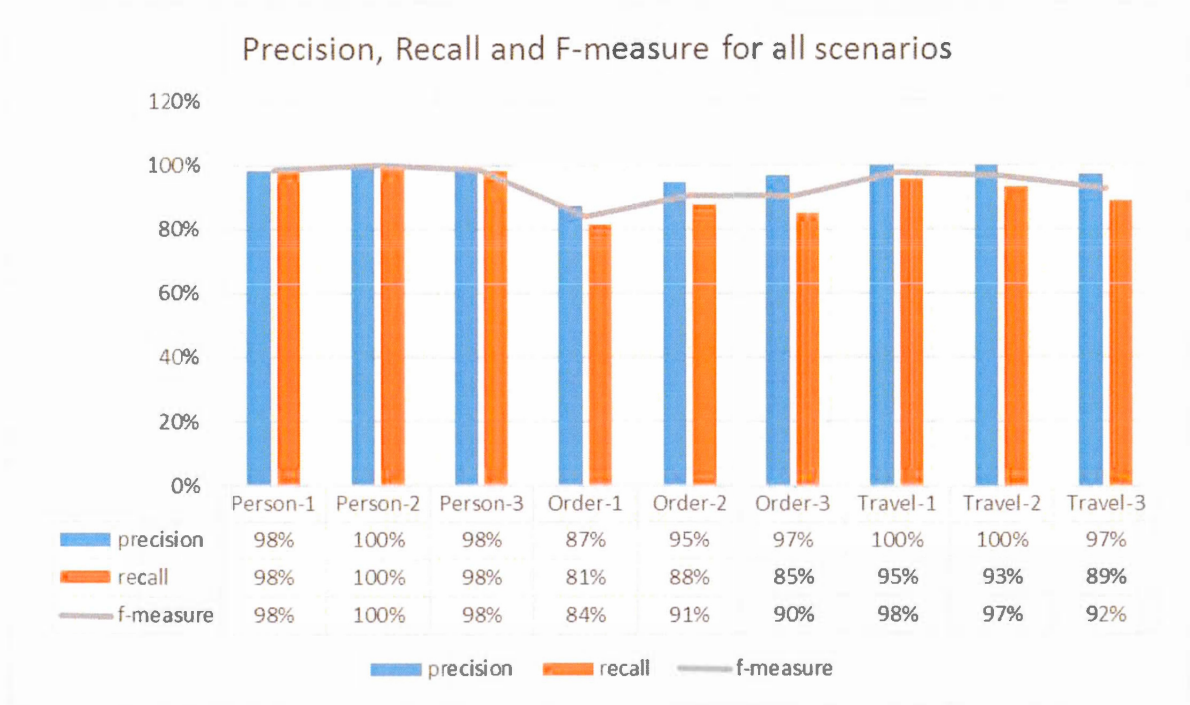


Figure 4.11 Précision, Rappel et F-Mesure pour tous les scénarios

Les moyennes des mesures de performance *précision*, *rappel* et *F-mesure*, que nous montrons au niveau des méta-simulations ne reflètent pas nécessairement la performance de ces dernières. En effet, la performance de la méta-simulation doit être calculé en fonction du nombre de correspondances correctement trouvées au niveau de la méta-simulation et non pas être une moyenne des performances obtenues au niveau des simulations individuelles.

Ainsi donc, nous avons compiler les résultats concernant la performance au niveau des méta-simulations pour tous les scénarios. Comme l'indique le tableau 4.7, nous constatons que notre outil a été capable de trouver correctement toutes les

correspondances attendues par l'utilisateur (un taux de réussite de 100%). Avec de tels résultats, il est incontestable de dire que notre prototype tient toutes ces promesses au niveau de la qualité de l'alignement (correspondances trouvées).

Scénario	méta-simulation no	Nbr. correspondances à trouver	Nbr. correspondances trouvées correctes	% correspondances trouvées correctes
Person-1	1	6	6	100%
Person-2	2	6	6	100%
Person-3	3	6	6	100%
Order-1	1	8	8	100%
Order-2	2	8	8	100%
Order-3	3	8	8	100%
Travel-1	1	15	15	100%
Travel-2	2	15	15	100%
Travel-3	3	15	15	100%

Tableau 4.7 Résultat combiné pour la qualité de l'alignement trouvé

La figure montre la comparaison entre le résultat trouvé et le résultat attendu pour tous les scénarios. On constate que notre prototype a réussi à trouver, lors de chaque méta-simulation, et pour chaque scénario, toutes les correspondances attendues par l'utilisateur avec un taux de réussite de 100%.

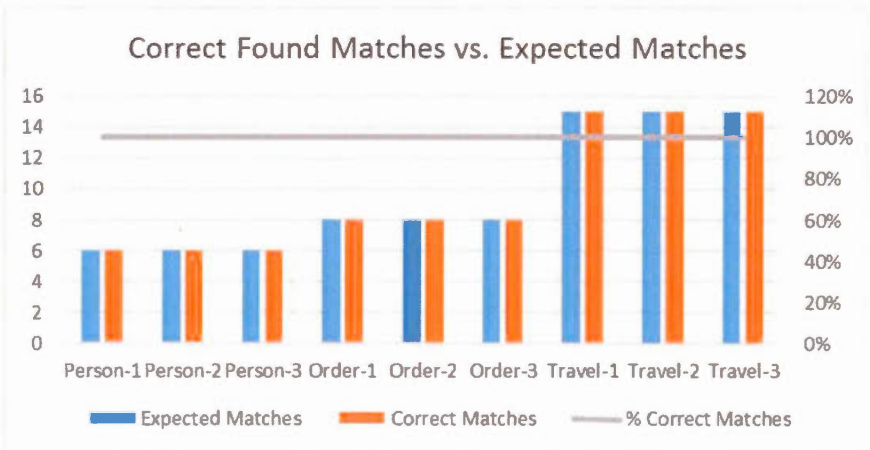


Figure 4.12 Comparaison des résultats pour tous les scénarios

Dans le but de relativiser « le parcours sans faute » de notre outil, nous avons été curieux de savoir jusqu'à quel point la performance, obtenue au niveau des méta-simulations, peut être impactée d'un côté par le nombre de simulations individuelles composant une méta-simulation et de l'autre côté par le nombre maximal d'itérations[105] d'une simulation individuelle (borne supérieure de la durée de vie d'une simulation).

Dans le but de valider l'impact du nombre de simulations individuelles d'une méta-simulation sur la qualité du résultat final, nous avons donc décidé de conduire une nouvelle expérimentation avec une réduction du nombre de simulations individuelles, composant une méta-simulation, à seulement 3 simulations individuelles au lieu de 10 simulations.

Ainsi que le met en lumière la figure 4.13, la performance a baissé pour le scénario « *Order* » durant les méta-simulations n° 2 et n° 3 ainsi que pour le scénario « *Travel* » durant la méta-simulation n° 1. Cela veut dire que l'outil n'a pas pu trouver correctement toutes les correspondances attendues durant ces méta-simulations. Sans contredit, nous pouvons conclure que le nombre de simulations individuelles, composant une méta-simulation, est un facteur important pour assurer une bonne performance de la part de notre outil (une meilleure quantification de l'incertitude concernant le résultat de l'appariement) surtout lorsqu'il s'agit de scénarios impliquant des schémas de grande taille et/ou ayant un niveau d'hétérogénéité élevé.

[105] Le paramètre *runLength* de la simulation en lot

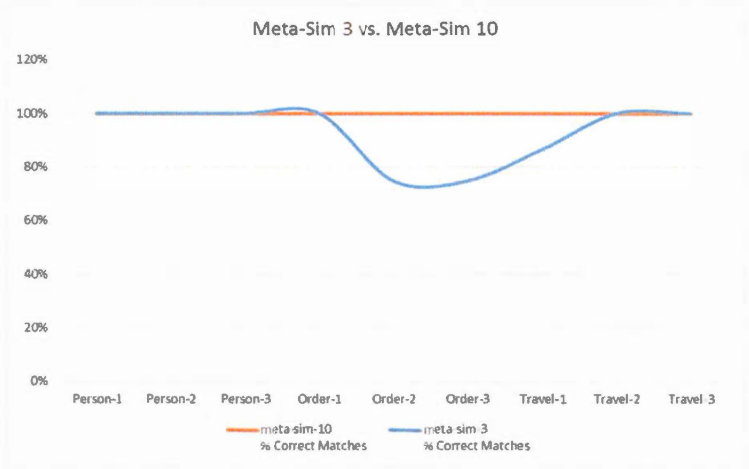


Figure 4.13 Impact du nombre de simulations sur la performance

Avec la preuve établie d'une corrélation entre le nombre de simulations individuelles composant une méta-simulation et la qualité du résultat final, nous nous sommes attelés à valider l'impact de la durée de vie des simulations individuelles sur la qualité du résultat final. Pour ce faire, nous avons conduit des expérimentations en réduisant à chaque fois la valeur du paramètre concernant la durée de la simulation « *runLength* » (en faisant baisser la valeur de *runLength* de 10.000 itérations à 3.000 itérations et ensuite à 300 itérations).

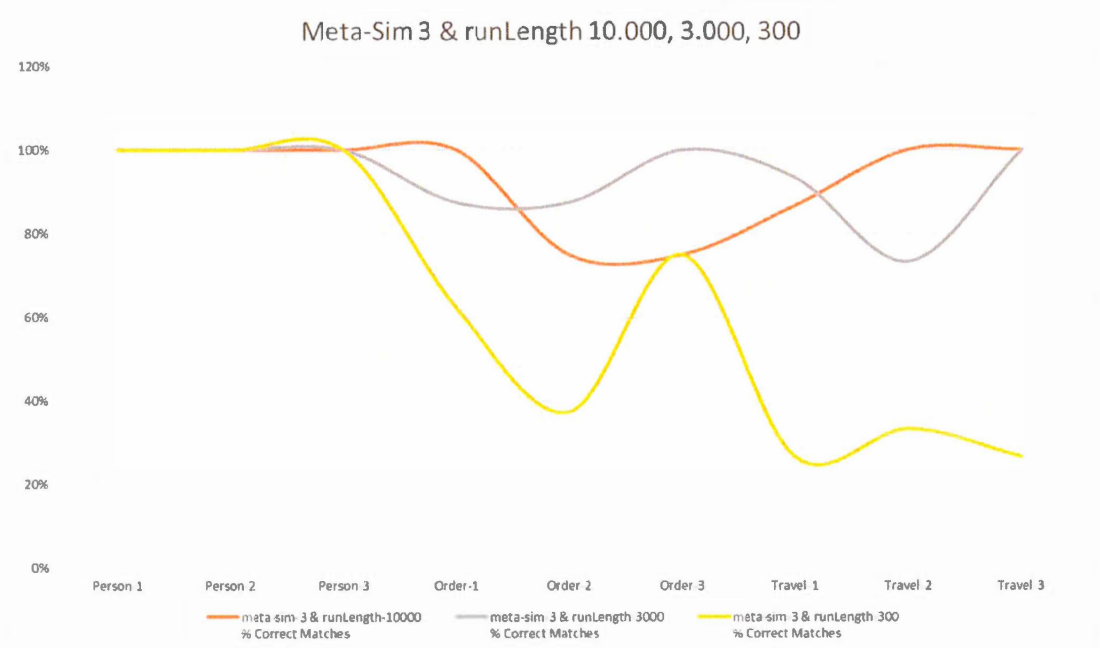


Figure 4.14 Impact du nombre d'itérations d'une simulation sur la performance

Comme le met en relief la figure 4.14, lorsque l'on observe la performance pour un scénario de difficulté basse comme le scénario « *Person* », nous constatons que cette performance ne se trouve pas impactée par la réduction de *runLength* (même avec une réduction de la durée des simulations à 300 itérations maximum). Or, lorsque l'on passe à des scénarios dont l'hétérogénéité lexicale ou la taille des schémas est considérée comme moyenne ou élevée, comme c'est le cas pour les scénarios « *Order* » et « *Travel* », nous constatons que la performance accuse une baisse très notable lorsque le *runLength* descend à une valeur assez basse comme 300. En fait, nous pouvons constater que l'impact de la réduction de la durée de vie des simulations, avec un *runLength* à 300, est encore plus marqué sur la qualité lorsqu'il s'agit d'un scénario avec des schémas de grande taille, comme c'est le cas pour le schéma « *Travel* ».

Avant d'avancer une explication à cette corrélation entre la performance et la durée des simulations, il serait pertinent de rappeler que la capacité de notre modèle à faire

émerger une solution optimale passe par des mécanismes, s'inscrivant dans une dimension temporelle, qui sont l'adaptation (au niveau micro), l'évolution et la coévolution (interactions agents-agents et agents-environnement au niveau macro).

Ainsi donc, on arrive à la déduction que la durée de vie des simulations (le nombre maximal d'itérations), tout comme le nombre de simulations composants une méta-simulation, est un facteur crucial à l'expression d'une évolution du système vers l'émergence d'une auto-organisation reflétant une solution optimale au problème de l'appariement.

L'aspect de l'efficacité ayant été démontré, venons-en maintenant à l'aspect de l'efficience. Il est important de souligner à ce propos, comme nous l'avions déjà anticipé, qu'aucun effort utilisateur, n'a été requis pour l'obtention d'un taux de succès aussi élevé. Rappelons brièvement que notre système d'appariement automatique se veut être un système adaptatif capable de s'ajuster automatiquement à tous les scénarios et à tous les changements. En effet, nous avons pu démontrer que malgré le changement des scénarios, provenant de divers domaines d'affaire, notre outil a pu s'adapter d'une manière autonome pour faire émerger, à chaque fois, les résultats attendus. Ainsi, nous estimons pouvoir confirmer que notre outil, d'appariement automatique de schémas *Reflex-SMAS,* ne requiert, de la part de l'utilisateur, pour trouver correctement les correspondances attendues, aucun effort de configuration ni d'optimisation.

Concernant le temps de réponse nécessaire pour l'obtention du résultat final, pour un scénario d'appariement (i.e. l'alignement), comme nous l'avons déjà mentionné, le résultat final est le fruit d'une analyse statistique sur une méta-simulation (pour nos expérimentations, nous avons fixé le nombre de simulations individuelles composant une méta-simulation à 10). Cela veut dire le temps de réponse correspond à l'exécution, en parallèle, de 10 simulations individuelles. Au niveau des simulations individuelles, nous nous intéressons à l'analyse du temps discret (i.e. itérations), qui représente en

quelques sortes la mesure de l'effort déployé par les agents pour former des appariements consensuels.

Sur ce point, la figure 4.15 présente le temps de réponse par rapport à chaque méta-simulation et chaque scénario. Nous pouvons, en examinant le graphique, déduire la corrélation existante entre le temps de réponse et la nature des différents scénarios. Nous pouvons par exemple voir que la taille des schémas (i.e. scénario « *Travel* ») ou encore l'hétérogénéité lexicale élevé (i.e. scénario « *Order* ») corrèle négativement avec le nombre d'itérations nécessaires pour trouver la solution au niveau de chaque méta-simulation. D'un autre côté, nous pouvons remarquer que le temps d'exécution (en minutes) des différentes méta-simulations pour les différents scénarios oscille, à peu près, entre 2 et 3 min.

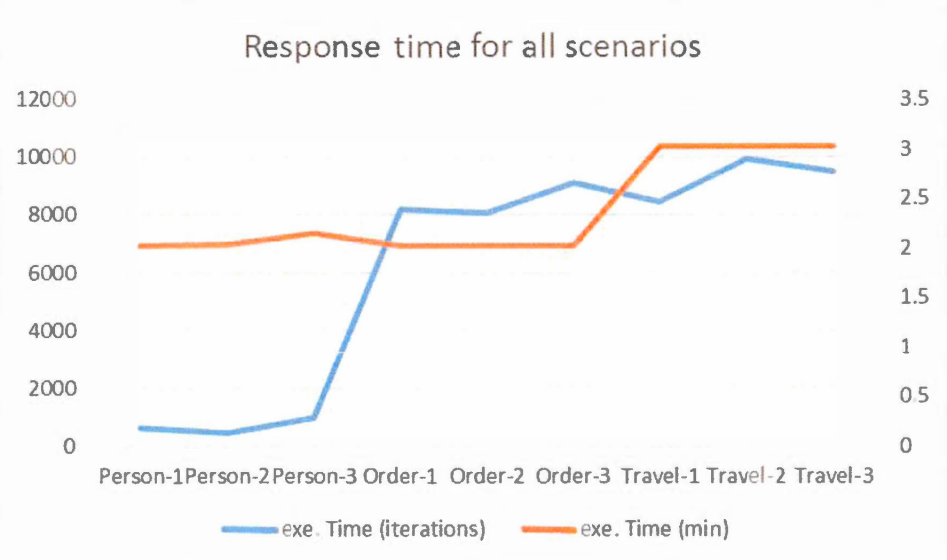


Figure 4.15 Temps de réponse pour tous les scénarios

Dans l'optique d'investiguer l'impact du nombre des simulations individuelles quant au temps total d'exécution de la méta-simulation, nous avons décidé d'analyser les

résultats obtenus lors de l'expérimentation faite antérieurement avec seulement 3 simulations individuelles.

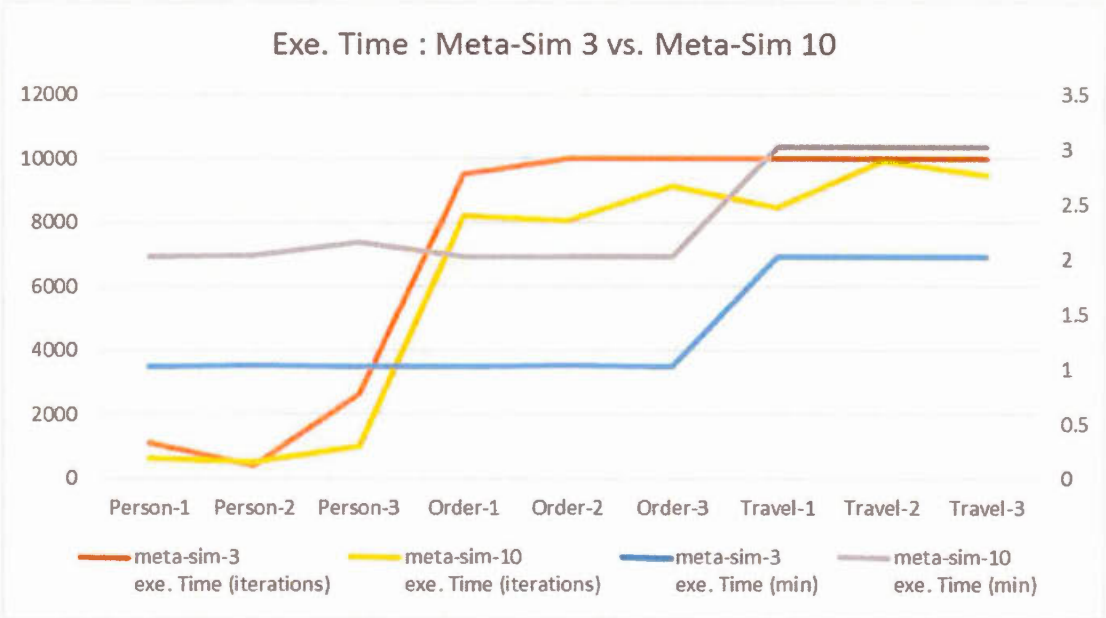


Figure 4.16 Impact du nombre des simulations le temps total d'exécution

Nous constatons que la moyenne du nombre d'itérations ne varie pas beaucoup selon le nombre de simulations individuelles composant une méta-simulation. Par contre, nous pouvons constater une baisse du temps total d'exécution pour les méta-simulations se composant de seulement 3 simulations individuelles. Nous pouvons donc conclure, que la baisse du nombre de simulations individuelles composant les méta-simulations corrèle positivement avec le temps total d'exécution pour ces dernières.

Dans la même veine, nous avons voulu explorer le rôle joué par le parallélisme dans la réduction du temps total d'exécution des méta-simulations. Pour ce faire, nous avons décidé de refaire les mêmes expérimentations, pour les différents scénarios, mais en prenant le soin, cette fois-ci, d'éliminer le parallélisme de l'équation. Nous avons donc

fait tourner les méta-simulations, pour les différents scénarios, en réduisant le nombre d'instances (i.e. machine virtuelle java), supportant l'exécution des simulations individuelles, de 10 instances à une seule instance.

Le résumé des résultats, illustré par la figure 4.17, nous montre clairement une nette augmentation du temps d'exécution avec l'élimination du parallélisme (une seule instance).

Figure 4.17 Impact du nombre d'instances sur le temps d'exécution

Nous pouvons donc conclure que la baisse du nombre d'instances supportant l'exécution des simulations individuelles, composant les méta-simulations, corrèle négativement avec le temps total d'exécution pour ces dernières.

Compte tenu de ces considérations, nous sommes d'avis que le temps d'exécution (une moyenne de 2.5 minutes, tout scénario confondu, pour une méta-simulation composée de 10 simulations), pour une méta-simulation, reste, tout compte fait, un prix raisonnable à payer pour l'obtention d'une meilleure qualité de l'alignement. Nous considérons donc, que pour certaines applications, nécessitant l'appariement

automatique de schémas, ce temps de réponses pourrait être considéré, dans une large mesure, comme étant dans les limites du viable et de l'acceptable.

Avant de conclure, il y aurait intérêt à contraster les résultats de notre prototype *Reflex-SMAS* avec ceux qui ont résulté des tests[106] effectués, à l'aide de l'outil *COMA,* pour l'évaluation de la pertinence des scénarios d'appariement conçus pour les expérimentations faisant l'objet de ce chapitre.

La tableau 4.8 présente une comparaison des performances obtenues, pour les scénarios « *Person* », « *Order* » et « *Travel* », avec notre prototype par rapport à celles obtenues avec l'outil *COMA*.

		Reflex-SMAS		**COMA**	
Scénario	Nbr. corresp. à trouver	Nbr. corresp. trouvées correctes	% corresp. trouvées correctes	Nbr. corresp. trouvées correctes	% corres. trouvées correctes
Person	6	6	100%	5	83%
Order	8	8	100%	6	75%
Travel	15	15	100%	13	87%

Tableau 4.8 Performances de *Refle-SMAS* par rapport à l'outil *COMA*

[106] Les tests ont exploité les stratégies (i.e. flux de travail) par défaut (i.e. *$NodesNameS* et *$NodesS*)

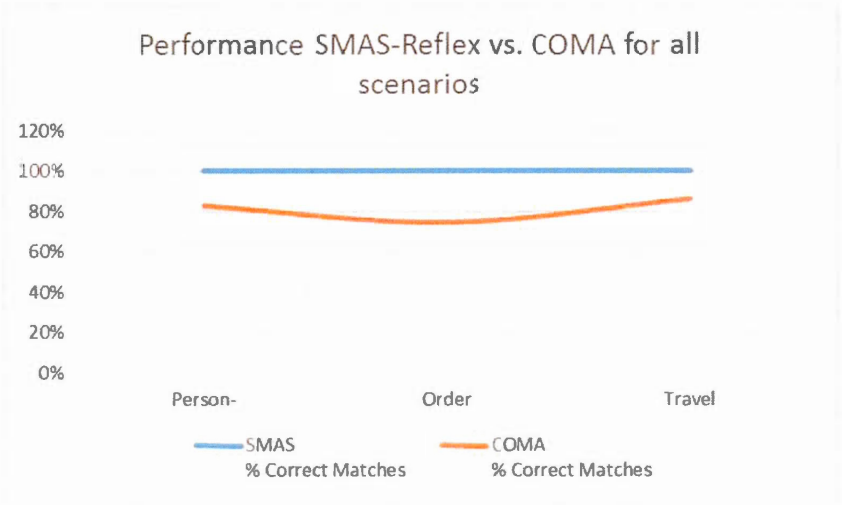


Figure 4.18 Performances de *Reflex-SMAS* par rapport à l'outil *COMA*

Les différentes questions que nous nous étions posées étant ainsi résolues, il nous reste à conclure que notre outil d'appariement de schémas affiche des performances exceptionnelles, pour les scénarios testés, et cela sans sacrifier le temps de réponse, qui reste dans les limites de l'acceptable.

Il est à noter, cependant, que l'architecture réflexive (règles simples) sur laquelle repose notre prototype *SEAgentReflex,* présente quelques limitations. Nous avons remarqué par exemple que lorsque l'on exécute notre prototype pour des schémas de tailles différentes, les agents en surplus dans l'un des deux schémas, et qui par conséquent ne peuvent pas former d'appariement consensuel, continuent malgré tout à chercher à former un appariement consensuel forçant par ce comportement naïf, la simulation à durer jusqu'à l'atteinte du nombre maximal d'itérations.

4.5 Conclusion

De toutes ces considérations, nous pouvons confirmer que la preuve a été apportée pour la viabilité de notre approche, permettant de tenir en compte la réalité systémique de l'appariement automatique de schémas et ce en l'abordant sous l'angle des systèmes complexes adaptatifs, et plus spécifiquement en le modélisant en une simulation multi-agents.

Nous estimons devoir insister sur le fait qu'au meilleur de nos connaissances, jamais l'appariement automatique de schémas n'a été abordé en adoptant la *pensée systémique* (*holistique*) de manière à appréhender toutes sa complexité intrinsèque.

Plus qu'un nouvel outil d'appariement, *Reflex-SMAS*, représente un changement de paradigme, pour l'appariement automatique de schémas.

CHAPITRE V

VERS UNE SIMULATION D'AGENTS RATIONNELS (RATIONAL-SMAS)

5.1 Introduction

Ce chapitre se penche sur une proposition de l'évolution de l'architecture interne de l'agent *SE-Agent*, de notre modèle de simulation multi-agents pour l'appariement de schémas (*SMAS)*, d'un agent de type réflexif vers un agent de type rationnel. La nouvelle version du prototype pouvant découlée de cette évolution pourrait être baptisée : ***Simulation d'agents rationnels pour l'appariement de schémas*** (*Rational-SMAS*).

5.2 Agent Rationnel (*SEAgentRational*)

Le but de cette proposition, étant de jeter la lumière sur les perspectives d'évolution possibles, pour notre prototype *ReflexSMAS* (basé sur les agents réflexifs). En effet, nous pensons que l'exploration de la possibilité de faire évoluer l'agent *réflexif* vers un agent *rationnel,* pouvant avoir un comportement basé sur le raisonnement et la délibération quant au choix des actions, peut être envisagée comme solution pour surpasser les limites d'un comportement purement réflexif.

L'intérêt de proposer une évolution de notre modèle conceptuel, d'un modèle réflexif vers un modèle rationnel, à notre sens, s'inscrit dans une logique de recherche de nouvelles solutions pour pallier aux lacunes observées lors de la validation de notre prototype *Reflex-SMAS* (liée à l'architecture interne des agents).

Les choix qui ont guidé la conception de ce nouveau modèle rationnel ont été influencés par les considérations suivantes :

- Respecter un des fondamentaux du paradigme des systèmes complexes adaptatifs, à savoir l'adoption de règles simples pour l'implémentation du comportement des agents.
- Veiller à ce que le changement d'un comportement réflexif vers un comportement rationnel ne pénalise pas la performance des simulations.

Ainsi, nous pensons que l'architecture qui doit être retenue pour les agents se doit d'être une architecture permettant des raisonnements simples du type d'une architecture d'agent rationnel délibératif.

5.2.1 Modèle conceptuel

Dans l'objectif de dépasser les limites d'un comportement purement réflexif, nous avons décidé d'explorer la possibilité de faire évoluer l'agent *réflexif* vers un agent *rationnel* pouvant avoir un comportement basé sur le raisonnement et la délibération quant au choix des actions à prendre.

L'agent *rationnel* quel nous visons devrait posséder une mémoire, une représentation partielle de son environnement et des autres agents (sa perception), ainsi qu'une capacité de raisonnement lui permettant de faire un choix rationnel (faire le choix de l'action avec la plus grande utilité) qui peut lui garantir de maximiser sa satisfaction (mesure de performance). La mesure de performance doit être un critère objectif de succès pour un comportement (e.g. la mesure de performance pour notre agent *Rational-SMAS* est sa capacité de trouver chez le groupe opposé, en un temps raisonnable, le bon appariement ou, s'il n'existe aucun appariement possible, d'abandonner le processus.

Dans la littérature, on parle aussi d'agents intentionnels, délibératifs ou cognitifs car ils possèdent des représentations explicites de leurs buts sur lesquelles ils sont capables de raisonner afin de produire des plans d'actions.

Notre agent rationnel, *Rational-SMAS,* s'approche de la classe d'agents à base d'utilité « *Utility-based Agent* » selon la classification de Russell et al. Comme la notion de bonheur de l'agent, qui ressort de la question « *how happy I will be in such a state* » (Figure 5.1 Agent à base d'utilité) n'a pas une connotation très scientifique, les économistes et les informaticiens préfèrent parler d'utilité (Russell *et al.*, 2010).

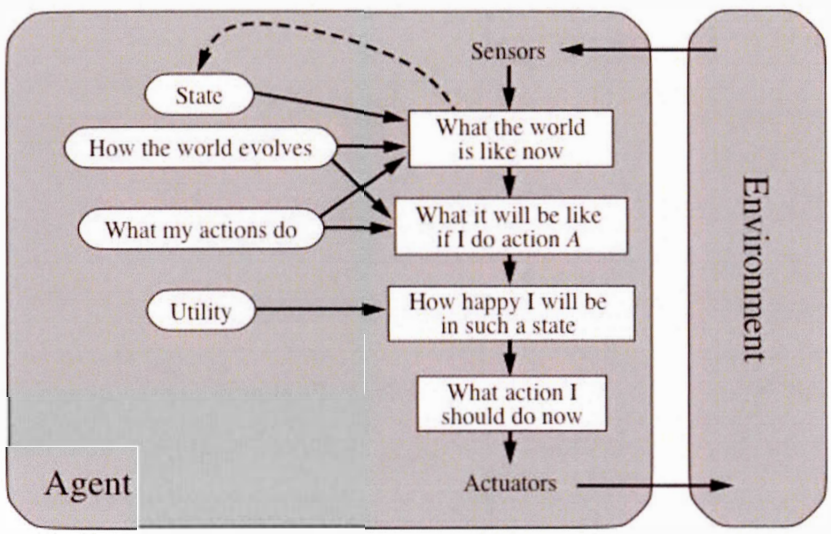


Figure 5.1 Agent à base d'utilité **(Russell *et al.*, 2010)**

En effet, notre agent prend ses décisions selon l'utilité maximum que peut lui procurer le choix de l'action. Pour ce faire, il utilise un diagramme de décision bayésien qui lui permet, après sa phase de perception, compte tenu des résultats des différents calculs de similarité ainsi que d'autres facteurs, basés sur sa propre perception de son environnements (les autres agents, déroulement de la simulation, etc.) et de son état, de raisonner sur le choix de la meilleure action à prendre.

Il est cependant très important de distinguer la rationalité de l'omniscience. Cette dernière est en réalité impossible car elle suppose que l'agent connait d'une manière exacte le résultat réel de ses actions (une sorte de perfection). En d'autres termes, la rationalité cherche à maximiser la performance attendue, alors que la perfection cherche à maximiser la performance réelle.

5.2.1.1 Perception

La phase de perception de notre agent rationnel, devrait étendre la logique de la phase de perception de notre agent générique (*SEAgent*) pour inclure :

- le calcul de similarité sémantique;
- le calcul des attributs de classification pour la prédiction du meilleur appariement (apparieur prédictif);
- la mise à jour de la mémoire (matrice interne mise à jour à chaque itération et représentant les connaissances de l'agent sur les agents de la population opposée).

Concernant le calcul de similarité sémantique, l'agent va comparer la similarité sémantique entre la concaténation de son nom et de sa description et celle des autres agents. La méthode utilisée pour le calcul de similarité sémantique est la méthode *Explicit Semantic Analysis* (ESA) (Gabrilovich et Markovitch, 2007).

Pour ce qui est de l'étape de prédiction de l'appariement, l'agent va exécuter le classificateur bayésien (*Weka API*) pour tenter de prédire l'agent du groupe opposé représentant le meilleur appariement. Le classificateur reçoit une matrice des agents à apparier avec les attributs mis à jour (extraite à partir de la mémoire). Cette matrice est utilisée par le classificateur pour créer les instances de test et faire la prédiction. La prédiction représentant le nom de l'agent qui est considéré comme le meilleur appariement est retournée à l'agent.

5.2.1.2 Décision

Comme pour l'agent réflexif, après la phase de perception vient la phase de décision. La différence majeure, entre l'agent rationnel et l'agent réflexif réside dans cette phase. En effet, à la différence de l'agent réflexif, l'agent rationnel ne se base pas sur de simples heuristiques (règles simples) pour prendre ses décisions mais se base sur un *modèle de prise de la décision sous l'incertitude* implémenté sous la forme d'un réseau de décision bayésien *(diagramme d'influence*[107]*)*.

La figure 5.2 montre le diagramme d'influence utilisé par l'agent rationnel lors de la phase de décision. Ce diagramme d'influence a été développé en utilisant l'outil *UnBBayes*[108] (Matsumoto *et al.*, 2011).

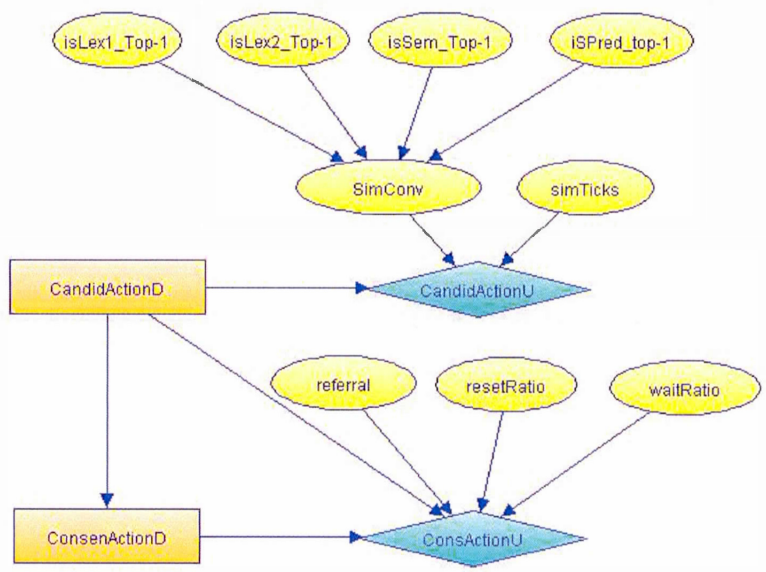


Figure 5.2 Diagramme d'influence pour l'agent *SEAgentRational*

[107] De l'anglais « *influence diagram* »

[108] Outil Java à code source ouvert

Notre diagramme d'influence se compose des nœuds suivants :

- Nœud de chance *isLex1_Top-1* : les états associés à ce nœud sont *true* et *false*. Ce nœud prend la valeur vraie si l'agent du groupe opposé a le meilleur score de similarité lexicale au niveau du nom
- Nœud de chance *isLex2_Top-1* : les états associés à ce nœud sont *true* et *false*. Ce nœud prend la valeur vraie si l'agent du groupe opposé a le meilleur score de similarité lexicale au niveau du commentaire
- Nœud de chance *isSem_Top-1* : les états associés à ce nœud sont *true* et *false*. Ce nœud prend la valeur vraie si l'agent du groupe opposé a le meilleur score de similarité sémantique pour le résultat de la concaténation du nom et du commentaire
- Nœud de chance *isPred_Top-1* : les états associés à ce nœud sont *true* et *false*. Ce nœud prend la valeur vraie si l'agent du groupe opposé est prédit comme le meilleur appariement par l'apparieur prédictif
- Nœud de chance *SimConv* : les états associés à ce nœud sont *none, low* et *high*. Ce nœud prend un des états associés selon les états des nœuds de chance parents. La valeur *none* et *low* représente respectivement une convergence, des calculs de similarité, nulle et faible vers un appariement candidat et la valeur *high* représente une convergence haute vers un appariement candidat
- Nœud de chance *SimTicks* : les états associés à ce nœud sont *none, low* et *high*. Ce nœud représente l'évolution de la simulation dans le temps
- Nœud de décision *CandidActionD* : les actions associées à ce nœud sont *explore, exploit* et *abort*. Ce nœud représente les actions à prendre en fonction du calcul d'utilité pour chaque action par le nœud d'utilité (*CandidActionU)*
- Nœud d'utilité *CandidActionU* : la table associée à ce nœud représente un calcul d'une fonction d'utilité les actions, *explore, exploit* et *abort*, associés au nœud de décision (*CandidActionD*) en fonction du degré de convergence de

similarité (*SimConv*) et du stade de l'évolution de la simulation dans le temps (*SimTicks*)

- Nœud de chance *referral* : les états associés à ce nœud sont *true, false*. Ce nœud prend la valeur vraie si l'agent du groupe opposé a choisi l'agent comme un appariement candidat
- Nœud de chance *resetRatio* : les états associés à ce nœud sont *none, low* et *high.* Ce nœud représente le ratio du nombre de réinitialisation des croyances sur l'appariement candidat par rapport au déroulement de la simulation
- Nœud de chance *waitRatio* : les états associés à ce nœud sont *none, low* et *high.* Ce nœud représente le ratio du temps d'attente d'un appariement consensuel par rapport au déroulement de la simulation
- Nœud de décision *ConsenActionD* : les actions associées à ce nœud sont *exploit*, *wait*, *reset* et *abort*. Ce nœud représente les actions à prendre en fonction du calcul d'utilité pour chaque action par le nœud d'utilité (*ConsenActionU)*
- Nœud d'utilité *ConsenActionU* : la table associée à ce nœud représente un calcul d'une fonction d'utilité les actions à prendre (*ConsenActionD)* en fonction de la décision précédente (*CandidActionD*), du ratio de l'attente (*waitRatio*), du ratio des réinitialisations (*resetRatio*) et de l'état du nœud *referral*

L'agent *SEAgentRational* va utiliser ce diagramme d'influence pour la prise de décisions concernant la sélection d'un appariement candidat ainsi que la prise de décision concernant la sélection d'un appariement consensuel.

Concernant la sélection d'un appariement candidat, avec le résultat des différents calculs lors de la phase de l'appariement, l'agent rationnel va, pour chaque agent du groupe opposé, mettre à jour les variables aléatoires (nœuds de chance) du diagramme d'influence et ensuite *propager les croyances* dans le but de capturer le score de

convergence pour chaque agent du groupe opposé. Pour l'agent du groupe opposé ayant le meilleur score de convergence, l'agent rationnel, va interroger le diagramme d'influence pour sélectionner l'action à effectuer (i.e. *explore*, *exploit* ou *abort*).

Concernant la sélection d'un appariement consensuel, l'agent rationnel va lors de la phase de décision de chaque itération, évaluer la possibilité de sélectionner l'appariement candidat comme un appariement consensuel. Pour ce faire une mise à jour de la variables aléatoire *referral* (nœuds de chance) du diagramme d'influence est nécessaire avant de *propager les croyances* dans le but d'interroger le diagramme d'influence pour la sélection de l'action à effectuer (i.e. *exploit*, *wait*, *reset* ou *abort*). La variables aléatoire *referral* indique si oui ou non l'agent, du groupe opposé, représentant l'appariement candidat a lui aussi choisi l'agent comme appariement candidat (capture d'évènement venant de l'environnement).

5.2.1.3 Action

Les actions que l'agent peut sélectionner à partir du digramme d'influence à l'étape de la décision sur l'action à prendre pour la sélection d'un appariement candidat sont : *explore*, *exploit* ou *abort*. L'action *explore* veut dire que l'agent doit continuer à explorer par ce que l'agent, du groupe opposé, avec le meilleur score de convergence ne devrait pas être choisi comme appariement candidat. L'action *exploit* par contre veut dire que l'agent doit sélectionner l'agent, du groupe opposé, avec le meilleur score de convergence comme appariement candidat. Et enfin, L'action *abort* veut dire que l'agent doit se retirer de la simulation car il n'a pas de chances de trouver un appariement candidat (prend en considération la variable aléatoire *simTick* qui donne une idée sur le niveau d'avancement de la simulation dans le temps).

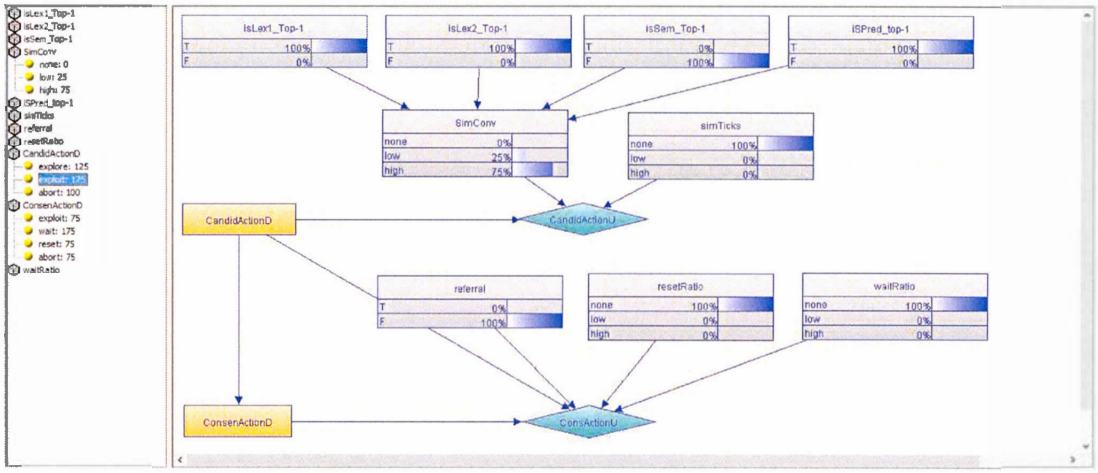


Figure 5.3 Propagation des croyances pour l'appariement candidat

Les actions que l'agent peut sélectionner, à partir du digramme d'influence, à l'étape de la décision sur l'action à prendre pour la sélection d'un appariement consensuel sont : *exploit*, *wait*, *reset* ou *abort*. L'action *exploit* veut dire que l'agent doit sélectionner l'agent, représentant l'appariement candidat, comme un appariement consensuel. L'action *reset*, par contre, veut dire que l'agent doit effacer ses croyances concernant l'appariement candidat. L'action *wait* veut dire que l'agent doit attendre que l'agent représentant l'appariement candidat le choisisse à son tour comme l'appariement candidat. Et enfin, L'action *abort* veut dire que l'agent doit se retirer de la simulation car il n'a pas de chances de trouver un appariement consensuel (prend en considération les variables aléatoires *waitRatio,* donne une idée sur le ratio du temps d'attente d'un appariement consensuel par rapport au déroulement de la simulation, et la variable et *resetRatio* qui donne une idée sur le taux de réinitialisation des croyances sur l'appariement candidat tout au long de la simulation).

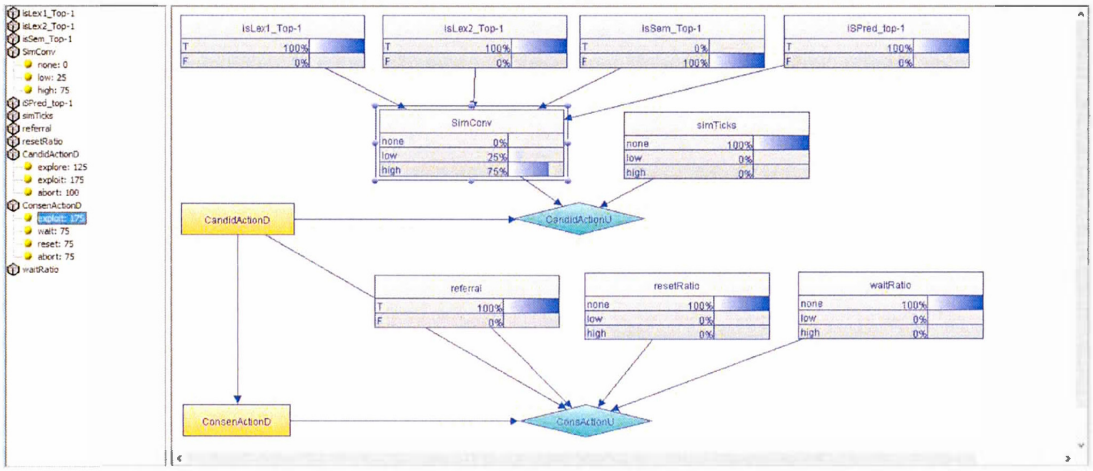


Figure 5.4 Propagation des croyances pour l'appariement consensuel

5.2.2 Architecture et implémentation

Pour cette version du prototype, *Rational-SMAS*, la simulation va référencer la classe *SEAgentRational* (implémentant la classe abstraite la classe *SEAgent).*

Le comportement de l'agent représenté par cette classe (i.e. *SEAgentRational*) peut être décrit par le diagramme d'activités ci-dessous (figure 5.5).

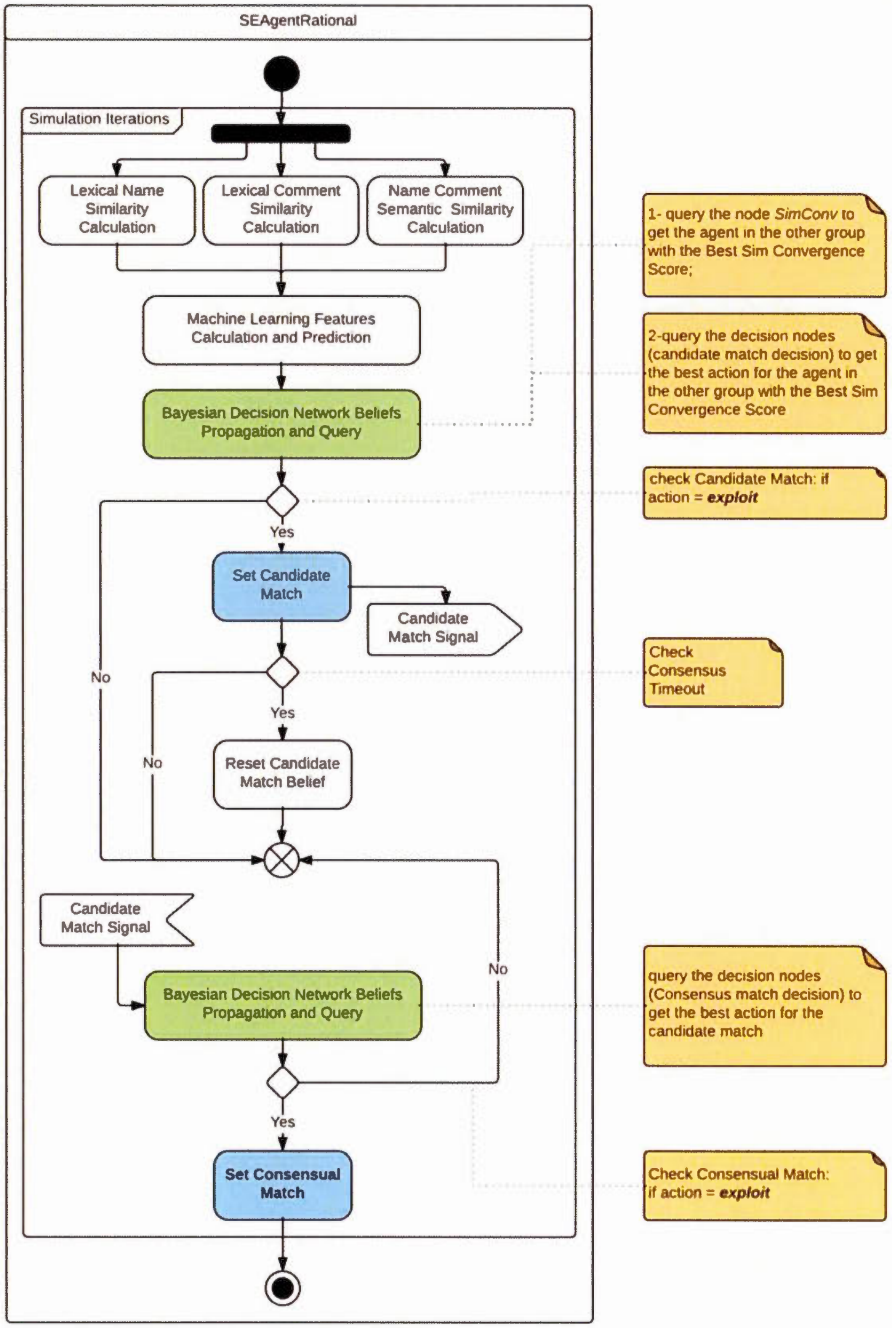


Figure 5.5 Diagramme d'activités pour l'agent SEAgentRational

5.3 Conclusion

Dans ce chapitre nous avons fait la proposition, sur le plan « conceptuel », d'une évolution de l'architecture interne de l'agent *SE-Agent*, de notre modèle de simulation multi-agents pour l'appariement de schémas (*SMAS)*, d'un agent de type réflexif vers un agent de type rationnel.

Cette proposition vise à ouvrir la voie à une l'implémentation et à une validation d'une évolution possible, pour notre prototype *ReflexSMAS* (basé sur les agents réflexifs) qui pourrait donner lieu à une comparaison des performances des deux architectures.

CONCLUSION

L'appariement automatique de schémas est une tâche complexe à plus d'un titre : (i) l'hétérogénéité et l'ambiguïté intrinsèque aux éléments de schémas à apparier, (ii) le caractère incertain des résultats de l'appariement, (iii) le défi que peut poser l'optimisation de l'appariement (explosion combinatoire), etc.

De par leur origine commune, *la pensée réductionniste*, les systèmes classiques, pour l'appariement automatique de schémas, exhibent les mêmes caractéristiques fondamentales et intrinsèque, qui participent à faire d'eux des systèmes *compliqués* (et non *complexes*) incapable de proposer des solutions simples et innovantes. Parmi ces caractéristiques fondamentales on trouve : la *linéarité* (les effets « extrants » produits sont proportionnels aux causes « intrants »), le *déterminisme* (en partant des mêmes intrants on obtient, avec un comportement *monotone* et *prévisible,* les mêmes extrants) et la *centralisation* (contrôle central pour le traitement des intrants et la production des extrants).

Dans le cadre de notre recherche nous avons investigué le recours à la théorie des systèmes complexes adaptatifs, issues de la pensée systémique, pour chercher, loin des sentiers battus, des réponses innovantes aux défis auxquels les approches classiques, d'appariement automatique de schémas, font toujours face (e.g. complexité, incertitude).

Nous avons donc proposé (chapitre 2) une approche découlant de la *pensée systémique,* en la situant, plus précisément, dans le domaine des *systèmes complexe adaptatifs*. Cette approche se veut être une approche de *simulation multi-agents* basée sur un modèle *non déterministe*, *stochastique* (*aléatoire*), *non prévisible* et *sans contrôle*

centrale, où chaque simulation peut donner lieu, même en partant des mêmes intrants, à des extrants différents. La multiplication des simulations permet l'élargissement de l'espace de possibilités et par voie de conséquence l'augmentation des potentialités d'une meilleure qualité du résultat de l'appariement (i.e. réduction de l'incertitude sur l'alignement obtenu).

Nous avons, à travers un modèle conceptuel pour l'appariement automatique de schémas, basé sur le paradigme de la *modélisation et de la simulation multi-agents*, cristallisé les principes phares de la pensée *systémique* et de la *théorie des systèmes complexes adaptatifs*, notamment *la non-linéarité*, *la stochasticité*, *l'auto-organisation* et *l'émergence*. Un modèle conceptuel qui formalise notre vision de l'appariement de schémas sous la perspective d'une simulation des interactions de deux groupes d'agents (réactifs) où chaque groupe représente un des deux schémas à apparier, et où les agents représentent les éléments de ces schémas, et où les comportements et les interactions entre ces agents représentent la recherche et la sélection des relations de correspondances.

Nous avons ensuite (chapitre 3), procédé à la transcription de ce modèle conceptuel en un modèle opérationnel (computationnel) qui a pris la forme, au terme d'une phase de développement, d'un prototype, *Reflex-SMAS,* tournant sur la plateforme de simulation *Repast Simphony*. Par ailleurs, à la suite des expérimentations préliminaires de notre prototype, nous avons relevé l'importance de la collecte de données lors des simulations en donnant des exemples concrets d'exploitation de ces données (e.g. réseau des interactions pour l'analyse de la performance des apparieurs ou encore la conception d'un apparieur prédictif basé sur un classificateur bayésien).

Le prototype *Reflex-SMAS* a été soumis à une série d'expérimentations (chapitre 4) dont l'objectif était de démontrer la viabilité de notre approche relativement à deux aspects principaux : (i) *l'efficacité* (augmentation de la qualité de l'alignement trouvé) et (ii) *l'efficience* (réduction de l'effort nécessaire à cette efficacité).

Les résultats obtenus lors de ces expérimentations, sont venus apporter la preuve, de la viabilité de notre approche, que ce soit sur le plan de *l'efficacité* ou encore sur celui de *l'efficience*. Avec la preuve de la viabilité apporté nous pouvons désormais annoncer la naissance d'un nouvel outil d'appariement de schémas d'un nouveau genre représentant un changement de paradigme, dans le domaine de l'appariement automatique de schémas (au meilleur de nos connaissances, jamais l'appariement automatique de schémas n'a été abordé en adoptant la *pensée systémique* (*holistique*) en le considérant comme un système complexe adaptatif et en le modélisant comme une simulation multi-agents).

Dans l'optique d'ouvrir de nouvelles perspectives (chapitre 5), nous avons proposé une évolution sur le plan « conceptuel » de l'architecture interne de notre agent *SEAgent,* le faisant évoluer d'un agent de type réactif (baptisé *SEAgentReflex*) à un agent de type rationnel (baptisé *SEAgentRational*). Cette évolution consiste en l'implémentation d'un *modèle de prise de la décision sous l'incertitude*, au niveau de la *phase de décision* de l'agent *SEAgentRational* lui conférant la capacité de raisonner et de faire le choix entre des actions conflictuelles. La résultante de cette évolution conceptuelle pourrait donner lieu à une nouvelle version de notre prototype (que l'on pourra baptisée *Rational-SMAS).*

Désormais, nous savons que notre approche est une approche prometteuse ; il importe maintenant de se tourner vers le futur et de présenter les points qui, selon notre point de vue, mériteraient que l'on s'y attarde pour améliorer notre modèle de simulation multi-agents pour l'appariement automatique de schémas. Ainsi, les travaux futurs, que nous estimons importants pour l'évolution de notre prototype *Reflex-SMAS*, peuvent être déclinés, par ordre de priorité, comme suit :

1. La conception d'une interface utilisateur et l'automatisation de tâches d'import de schémas (en entrée) et de génération de l'alignement (en sortie);
2. l'amélioration de la performance (i.e. temps de réponse);

3. l'exploitation de la structure des schémas (i.e. similarité structurelle);
4. l'introduction de nouvelle mesures de similarité (e.g. *google similarity measure*);
5. l'optimisation de notre apparieur prédictif;
6. l'implémentation et l'expérimentation de l'évolution conceptuelle concernant l'agent rationnel (i.e. *Rational-SMAS)*
7. la prise en charge de la problématique des appariements complexes (i.e. appariements de cardinalité n: m);
8. la prise en charge de la problématique de l'appariement à large échelle;
9. la généralisation de notre approche à l'appariement des ontologies.

En guise de conclusion, nous nous permettons de réitérer que notre approche, qui relève du courant de la pensée systémique et qui se situe dans la ligné des solutions issues du domaine des systèmes complexes adaptatifs, vient éclairer d'un œil nouveau le domaine de l'appariement automatique de schémas, et représente, en ce sens, un changement de paradigme significatif dans ce domaine.

BIBLIOGRAPHIE

Andrus, D. C. (2005). The wiki and the blog: Toward a complex adaptive intelligence community. *Studies in Intelligence*, *49*(3).

Arenas, M. et Libkin, L. (2008). XML data exchange: consistency and query answering. *Journal of the ACM (JACM)*, *55*(2), 7.

Assoudi, H. et Lounis, H. (2011). Self-Healing Data Exchange Process under Evolving Schemas: A New Mapping Adaptation Approach Based on Self-Optimization. Dans T. M. Khoshgoftaar (dir.), *13th IEEE International Symposium on High-Assurance Systems Engineering, HASE 2011, Boca Raton, FL, USA, November 10-12, 2011* (p. 188–190). IEEE Computer Society.

Assoudi, H. et Lounis, H. (2014a). A Multi-Agent-Based Approach for Autonomic Data Exchange Processes. Dans M. Reformat (dir.), *The 26th International Conference on Software Engineering and Knowledge Engineering, Hyatt Regency, Vancouver, BC, Canada, July 1-3, 2013* (p. 334–337). Knowledge Systems Institute Graduate School.

Assoudi, H. et Lounis, H. (2014b). Agent-based Stochastic Simulation of Schema Matching. Dans M. Reformat (dir.), *The 26th International Conference on Software Engineering and Knowledge Engineering, Hyatt Regency, Vancouver, BC, Canada, July 1-3, 2013* (p. 748–749). Knowledge Systems Institute Graduate School.

Assoudi, H. et Lounis, H. (2014c). Towards a Self-Organized Agent-Based Simulation Model for Schema Matching. *International Science Index, Knowledge and Innovation Sciences 2, 2014*.

Assoudi, H. et Lounis, H. (2014d). Towards an Agent-Based Simulation Model for Schema Matching. Dans *Eighth IEEE International Conference on Self-Adaptive and Self-Organizing Systems, SASO 2014, London, United Kingdom, September 8-12, 2014* (p. 197–198). IEEE.

Assoudi, H. et Lounis, H. (2015a). Coping with Uncertainty in Schema Matching: Bayesian Networks and Agent-Based Modeling Approach. Dans M. Benyoucef, M. Weiss, et H. Mili (dir.), *E-Technologies - 6th International Conference, MCETECH 2015, Montréal, QC, Canada, May 12-15, 2015, Proceedings* (Vol. 209, p. 53–67). Springer.

Assoudi, H. et Lounis, H. (2015b). Schema Matching as complex adaptive system. Dans *2015 10th International Conference on Intelligent Systems: Theories and Applications (SITA)* (p. 1-7).

Aumueller, D., Do, H.-H., Massmann, S. et Rahm, E. (2005). Schema and ontology matching with COMA++. Dans *Proceedings of the 2005 ACM SIGMOD international conference on Management of data* (p. 906–908). ACM.

Bär, D., Zesch, T. et Gurevych, I. (2013). DKPro Similarity: An Open Source Framework for Text Similarity. Dans *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics: System Demonstrations* (p. 121–126). Citeseer.

Bastian, M., Heymann, S., Jacomy, M. et others. (2009). Gephi: an open source software for exploring and manipulating networks. *ICWSM*, *8*, 361–362.

Bauer, B., Müller, J. P. et Odell, J. (2001). Agent UML: A formalism for specifying multiagent software systems. *International journal of software engineering and knowledge engineering*, *11*(03), 207–230.

Bellahsene, Z., Bonifati, A., Duchateau, F. et Velegrakis, Y. (2011). On evaluating schema matching and mapping. Dans *Schema matching and mapping* (p. 253–291). [s.l.] : Springer.

Bellahsene, Z., Bonifati, A. et Rahm, E. (2011). *Schema matching and mapping*, *20*. [s.l.] : Springer.

Bellahsene, Z. et Duchateau, F. (2011). Tuning for schema matching. Dans *Schema Matching and Mapping* (p. 293–316). [s.l.] : Springer.

Beni, G. (2009). Swarm Intelligence. Dans R. A. M. Ph. D (dir.), *Encyclopedia of Complexity and Systems Science* (p. 8869-8888). [s.l.] : Springer New York.

Bentley, G. (s.d.). *JoSQL (SQL for Java Objects)*. [s.l.] : [s.n.].

Berkovsky, S., Eytani, Y. et Gal, A. (2005). Measuring the relative performance of schema matchers. Dans *Web Intelligence, 2005. Proceedings. The 2005 IEEE/WIC/ACM International Conference on* (p. 366–371). IEEE.

Bernstein, P. A., Madhavan, J. et Rahm, E. (2011). Generic schema matching, ten years later. *Proceedings of the VLDB Endowment*, *4*(11), 695–701.

Bernstein, P. A., Melnik, S., Petropoulos, M. et Quix, C. (2004). Industrial-strength schema matching. *ACM SIGMOD Record*, *33*(4), 38–43.

Bertalanffy, L. von. (1968). *General system theory: Foundations, development, applications*. [s.l.] : Braziller. New York.

Bohannon, P., Elnahrawy, E., Fan, W. et Flaster, M. (2006). Putting context into schema matching. Dans *Proceedings of the 32nd international conference on Very large data bases* (p. 307–318). VLDB Endowment.

Bonabeau, E. (2002). Agent-based modeling: Methods and techniques for simulating human systems. *Proceedings of the National Academy of Sciences*, *99*(suppl 3), 7280–7287.

Bonami, M., Hennin, B. de et Boqué, J.-M. (1993). *Management des systèmes complexes: Pensée systémique et intervention dans les organisations*. [s.l.] : De Boeck Supérieur.

Bonifati, A., Mecca, G., Papotti, P. et Velegrakis, Y. (2011). Discovery and correctness of schema mapping transformations. Dans *Schema matching and mapping* (p. 111–147). [s.l.] : Springer.

Bourgine, P., Chavalarias, D. et Cohen-Boulakia, C. (2008). *Déterminismes et Complexités : du Physique à l'Ethique*. [s.l.] : La Découverte.

Castano, S., Ferrara, A., Lorusso, D., Näth, T. H. et Möller, R. (2008). *Mapping validation by probabilistic reasoning*. [s.l.] : Springer.

Castellani, B. (2014a). *Brian Castellani on the Complexity Sciences. Theory, Culture & Society*. Récupéré le 8 août 2015

Castellani, B. (2014b). *FOCUS: Complexity and the failure of quantitative social science. discoversociety.org*. Récupéré le 8 août 2015

Chaib-draa, B. et Dignum, F. (2002). Trends in agent communication language. *Computational intelligence*, *18*(2), 89–101.

Chapman, S. (2005). SimMetrics-open source Similarity Measure Library.

Choi, B. K. et Kang, D. (2013). *Modeling and Simulation of Discrete Event Systems*. [s.l.] : John Wiley & Sons.

Cohen, W., Ravikumar, P. et Fienberg, S. (2003). A comparison of string metrics for matching names and records. Dans *KDD Workshop on Data Cleaning and Object Consolidation* (Vol. 3, p. 73–78).

Cohen, W. W., Ravikumar, P. et Fienberg, S. (2003). Secondstring: An open source java toolkit of approximate string-matching techniques.

Cross, V. (2003). Uncertainty in the automation of ontology matching. Dans *Uncertainty Modeling and Analysis, 2003. ISUMA 2003. Fourth International Symposium on* (p. 135–140). IEEE.

Dhamankar, R., Lee, Y., Doan, A., Halevy, A. et Domingos, P. (2004). iMAP: discovering complex semantic matches between database schemas. Dans *Proceedings of the 2004 ACM SIGMOD international conference on Management of data* (p. 383–394). ACM.

Do, H.-H. et Rahm, E. (2002). COMA: a system for flexible combination of schema matching approaches. Dans *Proceedings of the 28th international conference on Very Large Data Bases* (p. 610–621). VLDB Endowment.

Do, H.-H. et Rahm, E. (2007). Matching large schemas: Approaches and evaluation. *Information Systems*, *32*(6), 857–885.

Doan, A., Domingos, P. et Halevy, A. (s.d.). Reconciling schemas of disparate data sources: A machine-learning approach (p. 509-520). ACM.

Doan, A., Domingos, P. et Halevy, A. Y. (2001). Reconciling schemas of disparate data sources: A machine-learning approach. Dans *ACM Sigmod Record* (Vol. 30, p. 509–520). ACM.

Doan, A., Domingos, P. et Levy, A. Y. (2000). Learning Source Description for Data Integration. Dans *WebDB (Informal Proceedings)* (p. 81–86).

Doan, A., Madhavan, J., Domingos, P. et Halevy, A. (2002). Learning to map between ontologies on the semantic web. Dans *Proceedings of the 11th international conference on World Wide Web* (p. 662–673). ACM.

Dooley, K. J. (1997). A complex adaptive systems model of organization change. *Nonlinear dynamics, psychology, and life sciences*, *1*(1), 69–97.

Duchateau, F. et Bellahsene, Z. (2014). Designing a benchmark for the assessment of schema matching tools. *Open Journal of Databases (OJDB)*, *1*(1), 3–25.

Duchateau, F., Coletta, R., Bellahsene, Z. et Miller, R. J. (2009). Yam: a schema matcher factory. Dans *Proceedings of the 18th ACM conference on Information and knowledge management* (p. 2079–2080). ACM.

Elmeleegy, H., Ouzzani, M. et Elmagarmid, A. (2008). Usage-based schema matching. Dans *Data Engineering, 2008. ICDE 2008. IEEE 24th International Conference on* (p. 20–29). IEEE.

Fagin, R., Haas, L. M., Hernández, M., Miller, R. J., Popa, L. et Velegrakis, Y. (2009). Clio: Schema mapping creation and data exchange. Dans *Conceptual Modeling: Foundations and Applications* (p. 198–236). [s.l.] : Springer.

Fagin, R., Kolaitis, P. G. et Popa, L. (2005). Data exchange: getting to the core. *ACM Transactions on Database Systems (TODS)*, *30*(1), 174–210.

Fishwick, P. A. (s.d.). COMPUTER SIMULATION: GROWTH THROUGH EXTENSION.

Foerster, H. von. (2007). *Understanding Understanding: Essays on Cybernetics and Cognition*. [s.l.] : Springer Science & Business Media.

Fortin, R. (2005). *Comprendre la complexité: introduction à La Méthode d'Edgar Morin*. [s.l.] : Presses Université Laval.

Gabrilovich, E. et Markovitch, S. (2007). Computing Semantic Relatedness Using Wikipedia-based Explicit Semantic Analysis. Dans *IJCAI* (Vol. 7, p. 1606–1611).

Gal, A. (2006a). Managing uncertainty in schema matching with top-k schema mappings. Dans *Journal on Data Semantics VI* (p. 90–114). [s.l.] : Springer.

Gal, A. (2006b). Why is schema matching tough and what can we do about it? *ACM Sigmod Record*, *35*(4), 2–5.

Gal, A. (2011a). Enhancing the capabilities of attribute correspondences. Dans *Schema Matching and Mapping* (p. 53–73). [s.l.] : Springer.

Gal, A. (2011b). Uncertain schema matching. *Synthesis Lectures on Data Management*, *3*(1), 1–97.

Gal, A., Anaby-Tavor, A., Trombetta, A. et Montesi, D. (2005). A framework for modeling and evaluating automatic semantic reconciliation. *The VLDB Journal—The International Journal on Very Large Data Bases*, *14*(1), 50–67.

Gal, A. et Shvaiko, P. (2009). Advances in ontology matching. Dans *Advances in web semantics i* (p. 176–198). [s.l.] : Springer.

Gale, D. et Shapley, L. S. (1962). College admissions and the stability of marriage. *American mathematical monthly*, 9–15.

Gintis, H. (2006). The economy as a complex adaptive system. *A review of Eric D. Beinhocker 'The Origins of Wealth: Evolution, Complexity, and the Radical Remaking of Economics',(accessed 02.10. 13.)*.

Giunchiglia, F., Shvaiko, P. et Yatskevich, M. (2004). S-Match: an algorithm and an implementation of semantic matching. Dans *ESWS* (Vol. 3053, p. 61–75). Springer.

Gong, J., Cheng, R. et Cheung, D. W. (2012). Efficient management of uncertainty in XML schema matching. *The VLDB Journal—The International Journal on Very Large Data Bases*, *21*(3), 385–409.

Grassé, P.-P. (1967). Nouvelles expériences sur le Termite de Müller (Macrotermes mülleri) et considérations sur la théorie de la stigmergie. *Insectes Sociaux*, *14*(1), 73-101.

Guedes, G. T. A. et Vicari, R. M. (2009). Applying AUML and UML 2 in the Multi-agent Systems Project. Dans *Advances in Conceptual Modeling-Challenging Perspectives* (p. 106–115). [s.l.] : Springer.

Holland, J. H. (2006). Studying complex adaptive systems. *Journal of Systems Science and Complexity*, *19*(1), 1–8.

Holmes, G., Donkin, A. et Witten, I. H. (1994). Weka: A machine learning workbench. Dans *Intelligent Information Systems, 1994. Proceedings of the 1994 Second Australian and New Zealand Conference on* (p. 357–361). IEEE.

Howard, R. A. (1984). The used car buyer problem. *The Principles and Applications of Decision Analysis*, *2*, 690–718.

Howard, R. A. et Matheson, J. E. (2005). Influence diagrams. *Decision Analysis*, *2*(3), 127–143.

Hu, W. et Qu, Y. (2008). Falcon-AO: A practical ontology matching system. *Web Semantics: Science, Services and Agents on the World Wide Web*, *6*(3), 237–239.

Huget, M.-P. (2004). Agent uml notation for multiagent system design. *Internet Computing, IEEE*, *8*(4), 63–71.

Jaccard, P. (1901). *Etude comparative de la distribution florale dans une portion des Alpes et du Jura*. [s.l.] : Impr. Corbaz.

Jones, W. (2003). Complex adaptive systems. *Guy Burgess and Heidi Burgess. Conflict Research Consortium, University of Colorado, Boulder. Posted: October*.

Laurencelle, L. (2001). *Hasard, nombres aléatoires et méthode Monte Carlo*. [s.l.] : PUQ.

Le Moigne, J.-L. (1990). La modélisation des systèmes complexes. *Paris: Bordas, Dunot, 1990*, *1*.

Lee, Y., Sayyadian, M., Doan, A. et Rosenthal, A. S. (2007). eTuner: tuning schema matching software using synthetic scenarios. *The VLDB Journal—The International Journal on Very Large Data Bases*, *16*(1), 97–122.

Levenshtein, V. I. (1966). Binary codes capable of correcting deletions, insertions and reversals. Dans *Soviet physics doklady* (Vol. 10, p. 707).

Li, J., Tang, J., Li, Y. et Luo, Q. (2009). Rimom: A dynamic multistrategy ontology alignment framework. *Knowledge and Data Engineering, IEEE Transactions on*, *21*(8), 1218–1232.

Macal, C. M. et North, M. J. (2006). Introduction to agent-based modeling and simulation. Dans *MCS LANS Informal Seminar*.

Macal, C. M. et North, M. J. (2009). Agent-based modeling and simulation. Dans *Winter simulation conference* (p. 86–98). Winter Simulation Conference.

Macal, C. M. et North, M. J. (2010). Tutorial on agent-based modelling and simulation. *Journal of Simulation*, *4*(3), 151–162.

Madhavan, J., Bernstein, P. A., Domingos, P. et Halevy, A. Y. (2002). Representing and reasoning about mappings between domain models. *AAAI/IAAI, 2002*, 80–86.

Madhavan, J., Bernstein, P. A. et Rahm, E. (2001). Generic schema matching with cupid. Dans *VLDB* (Vol. 1, p. 49–58).

Magnani, M. et Montesi, D. (2010). A survey on uncertainty management in data integration. *Journal of Data and Information Quality (JDIQ)*, *2*(1), 5.

Mao, M., Peng, Y. et Spring, M. (2008). A Harmony based Adaptive Ontology Mapping Approach. Dans *SWWS* (p. 336–342). Citeseer.

Marie, A. et Gal, A. (2008). Boosting schema matchers. Dans *On the Move to Meaningful Internet Systems: OTM 2008* (p. 283–300). [s.l.] : Springer.

Martinez-Gil, J., Alba, E. et Aldana-Montes, J. F. (2008). Optimizing ontology alignments by using genetic algorithms. Dans *Proceedings of the workshop on nature based reasoning for the semantic Web. Karlsruhe, Germany*.

Massmann, S. et Rahm, E. (2008). Evaluating Instance-based Matching of Web Directories. Dans *WebDB*. Citeseer.

Matsumoto, S., Carvalho, R. N., Ladeira, M., da Costa, P. C. G., Santos, L. L., Silva, D., Onishi, M., Machado, E. et Cai, K. (2011). UnBBayes: a java framework for probabilistic models in AI. *Java in Academia and Research. iConcept Press.*

McCann, R., AlShebli, B., Le, Q., Nguyen, H., Vu, L. et Doan, A. (2005). *Mapping maintenance for data integration systems*. [s.l.] : VLDB Endowment.

Melnik, S., Garcia-Molina, H. et Rahm, E. (2002). Similarity flooding: A versatile graph matching algorithm and its application to schema matching. Dans *Data Engineering, 2002. Proceedings. 18th International Conference on* (p. 117–128). IEEE.

Miller, R. J., Hernández, M. A., Haas, L. M., Yan, L.-L., Ho, C. H., Fagin, R. et Popa, L. (2001). The Clio project: managing heterogeneity. *SIgMOD Record*, *30*(1), 78–83.

Minar, N., Burkhart, R., Langton, C. et Askenazi, M. (1996). The swarm simulation system: A toolkit for building multi-agent simulations. Santa Fe Institute Santa Fe.

Monge, A. E. et Elkan, C. P. (1997). Efficient domain-independent detection of approximately duplicate database records. *Department of Computer Science and Engineering*, 92093–0114.

Ngo, D. et Bellahsene, Z. (2012). YAM++: a multi-strategy based approach for ontology matching task. Dans *Knowledge Engineering and Knowledge Management* (p. 421–425). [s.l.] : Springer.

Nian-Feng, W. et Xing-Chun, D. (2012). Uncertain Schema Matching Based on Interval Fuzzy Similarities. *International Journal of Advancements in Computing Technology*, *4*(1).

North, M. J. (2010). R and Repast Simphony.

North, M. J., Collier, N. T., Ozik, J., Tatara, E. R., Macal, C. M., Bragen, M. et Sydelko, P. (2013). Complex adaptive systems modeling with repast simphony. *Complex adaptive systems modeling*, *1*(1), 1–26.

North, M. J. et Macal, C. M. (2007). *Managing business complexity: discovering strategic solutions with agent-based modeling and simulation*. [s.l.] : Oxford University Press.

North, M. J., Tatara, E., Collier, N. T. et Ozik, J. (2007). *Visual agent-based model development with repast simphony*. Tech. rep., Argonne National Laboratory.

Nottelmann, H. et Straccia, U. (2007). Information retrieval and machine learning for probabilistic schema matching. *Information Processing & Management*, *43*(3), 552-576.

Odell, J., Parunak, H. V. D. et Bauer, B. (2000). Extending UML for agents. *Ann Arbor*, *1001*, 48103.

Paulheim, H. (2008). On applying matching tools to large scale ontologies. *OM-2008 held in conjunction with ISWC*, 214–218.

Pearl, J. (2011). Bayesian networks. *Department of Statistics, UCLA*.

Peukert, E. (2013). *Process-based Schema Matching: From Manual Design to Adaptive Process Construction*.

Peukert, E., Eberius, J. et Rahm, E. (2012). A self-configuring schema matching system. Dans *Data Engineering (ICDE), 2012 IEEE 28th International Conference on* (p. 306–317). IEEE.

Peukert, E., Massmann, S. et Koenig, K. (2010). Comparing Similarity Combination Methods for Schema Matching. *GI Jahrestagung (1)*, *10*, 692–701.

Pirró, G. et Talia, D. (2010). UFOme: An ontology mapping system with strategy prediction capabilities. *Data & Knowledge Engineering*, *69*(5), 444–471.

Rahm, E. (2011). Towards large-scale schema and ontology matching. Dans *Schema matching and mapping* (p. 3–27). [s.l.] : Springer.

Rahm, E. et Bernstein, P. A. (2001a). A survey of approaches to automatic schema matching. *the VLDB Journal*, *10*(4), 334–350.

Rahm, E. et Bernstein, P. A. (2001b). On matching schemas automatically. *VLDB Journal*, *10*(4), 334–350.

Ralph, D. S. (2000). *Strategic Management and Organisational Dynamics*. [s.l.] : Edinburgh: Person Education Limited.

Remondino, M. et Correndo, G. (2006). Mabs validation through repeated execution and data mining analisys. *International Journal of Simulation: Systems, Science & Technology*, *7*(6).

Ripley, B. D. (2001). The R project in statistical computing. *MSOR Connections*, *1*(1), 23–25.

Rizopoulos, N. (2010). *Schema Matching and Schema Merging based on Uncertain Semantic Mappings*. PhD Thesis, Imperial College London.

Russell, S. J., Norvig, P., Candy, J. F., Malik, J. M. et Edwards, D. D. (2010). *Artificial intelligence: a modern approach*. [s.l.] : Prentice hall.

Sayyadian, M., Lee, Y., Doan, A. et Rosenthal, A. S. (2005). Tuning schema matching software using synthetic scenarios. Dans *Proceedings of the 31st international conference on Very large data bases* (p. 994–1005). VLDB Endowment.

Schruben, L. W. (2015). *Don't Try These in the Real World.* [s.l.] : [s.n.].

Shannon, R. E. (1998). Introduction to the art and science of simulation. Dans *Proceedings of the 30th conference on Winter simulation* (p. 7–14). IEEE Computer Society Press.

Shoham, Y. (1993). Agent-oriented programming. *Artificial intelligence*, *60*(1), 51–92.

Shvaiko, P. et Euzenat, J. (2005). A Survey of Schema-Based Matching Approaches. Dans S. Spaccapietra (dir.), *Journal on Data Semantics IV* (p. 146-171). [s.l.] : Springer Berlin Heidelberg.

Shvaiko, P., Euzenat, J., Giunchiglia, F. et Stuckenschmidt, H. (2008). The 7th International Semantic Web Conference.

Siegfried, R. (2014). *Modeling and Simulation of Complex Systems: A Framework for Efficient Agent-Based Modeling and Simulation.* [s.l.] : Springer.

Simon, H. A. (1990). Sur la complexité des systèmes complexes. *Revue internationale de systémique*, *4*(2), 125–145.

Sklar, E. (2007). NetLogo, a multi-agent simulation environment. *Artificial life*, *13*(3), 303–311.

Steels, L. (2000). Language as a Complex Adaptive System. Dans M. Schoenauer, K. Deb, G. Rudolph, X. Yao, E. Lutton, J. J. Merelo, et H.-P. Schwefel (dir.), *Parallel Problem Solving from Nature PPSN VI* (p. 17-26). [s.l.] : Springer Berlin Heidelberg.

Stutz, J. et Cheeseman, P. (1994). A short exposition on Bayesian inference and probability. *NASA Ames Research Centre: Computational Sciences Division, Data Learning Group.*

Tian, A., Sequeda, J. F. et Miranker, D. P. (2013). QODI: Query as context in automatic data integration. Dans *The Semantic Web–ISWC 2013* (p. 624–639). [s.l.] : Springer.

Tisue, S. et Wilensky, U. (2004). Netlogo: A simple environment for modeling complexity. Dans *International Conference on Complex Systems* (p. 16–21).

Treuil, J.-P., Drogoul, A. et Zucker, J.-D. (2008). *Modélisation et simulation à base d'agents: exemples commentés, outils informatiques et questions théoriques.* [s.l.] : Dunod.

Velegrakis, Y., Miller, R. J. et Popa, L. (2003). Adapting mappings in frequently changing environments.

Velegrakis, Y., Miller, R. J. et Popa, L. (2004). Preserving mapping consistency under schema changes. *The VLDB Journal*, *13*(3), 274-293.

Villanyi, B., Martinek, P. et Szamos, A. (2014). Voting based fuzzy linguistic matching. Dans *2014 IEEE 15th International Symposium on Computational Intelligence and Informatics (CINTI)* (p. 27-32).

Wang, Z., Wang, Y., Zhang, S., Shen, G. et Du, T. (2006). Matching Large Scale Ontology Effectively. Dans R. Mizoguchi, Z. Shi, et F. Giunchiglia (dir.), *The Semantic Web – ASWC 2006* (p. 99-105). [s.l.] : Springer Berlin Heidelberg.

Wikipédia. (2015). Stigmergie. Dans *Wikipédia*. [s.l.] : [s.n.].

Winkler, W. E. (1990). String Comparator Metrics and Enhanced Decision Rules in the Fellegi-Sunter Model of Record Linkage.

Wise, M. J. (1996). YAP3: improved detection of similarities in computer program and other texts. Dans *ACM SIGCSE Bulletin* (Vol. 28, p. 130–134). ACM.

Yatskevich, M. (2003). Preliminary evaluation of schema matching systems.

Zhang, C. J., Chen, L., Jagadish, H. V. et Cao, C. C. (2013). Reducing uncertainty of schema matching via crowdsourcing. *Proceedings of the VLDB Endowment*, *6*(9), 757–768.

Zimmerman, B. (2001). Ralph Stacey's agreement & certainty matrix. *Schulich School of Business, York University, Toronto, Canada*.