Universitat Autònoma de Barcelona

A Pragmatic Approach to Translation:

Vocabulary Alignment through Multiagent Interaction and Observation

Doctoral Thesis

Paula Daniela Chocrón

Programa de Doctorat en Informàtica
Universitat Autònoma de Barcelona

Director i tutor: Marco Schorlemmer
Institut d'Investigació en Intel·ligència Artificial, CSIC

March 2018

This research was carried out at the Artificial Intelligence Research Institute (IIIA,CSIC). It was funded by the European Commission's Seventh Framework Programme (FP7/2007-2013) under grant agreement no. 607062 /ESSENCE: Evolution of Shared Semantics in Computational Environments/, and by CSIC's Proyectos Intramurales Especiales DIVERSIS (no. 201750E064) and SMA (no. 201550E040).

Abstract

Enabling collaboration between agents with different backgrounds is one of the objectives of open and heterogeneous multiagent systems. This can bring together participants with different knowledge, abilities, and access to resources, creating a richly open environment. For this collaboration to succeed, it needs to deal with different kinds of heterogeneity that can exist between agents. An important aspect of this heterogeneity is the linguistic one. To coordinate their collaborative actions, agents need to communicate with each other; and to ensure meaningful communication it is essential that they use the same vocabulary (and understand it in the same way).

The problem of achieving common understanding between agents that use different vocabularies has been mainly addressed by techniques that assume the existence of shared external elements, such as a meta-language, a physical environment, or semantic resources. These elements are not always available and, even when they are, they may yield alignments that are not useful for the particular type of interactions agents need to perform, as they are not contextualized.

In this dissertation we investigate a different approach to vocabulary alignment. We consider agents that only share knowledge of how to perform a task, given by the specification of an interaction protocol. We study the idea of *interaction-based vocabulary alignment*, a framework that lets agents learn a vocabulary alignment from the experience of interacting; by observing what works and what does not in a conversation. To give an intuition, consider someone trying to order a coffee in a foreign country. Even if there is no common language, the interaction is likely to succeed, since it consists of simple, well-understood steps that interlocutors agree on. Moreover, it is likely that, if our subject repeats the ordering coffee interaction many times, she will end up learning how it is performed in the foreign language. While humans are very good at adapting in this way, this idea has not been explored in depth for the case of artificial agents.

Throughout this dissertation we study how agents can learn a new vocabulary when they follow specifications that use different formalizations. Concretely, we consider interaction-based vocabulary alignment for protocols specified with finite state machines, with logical constraints, and with a social semantics based

on commitments. For each case, we provide techniques to infer semantic information from interacting, or observing interactions between other agents. We also analyze how these techniques can be used in combination with external alignments obtained in a different way. When these alignments are not necessarily correct, our techniques provide ways of repairing them.

For each type of specification we evaluate the proposed methods by simulating their use in a set of artificial, randomly generated protocols. This provides a general evaluation that does not suffer the biases of particular datasets. Later, we study how to apply our methods to an empirical dataset of human-crafted instructional protocols, obtained from the WikiHow webpage. We discuss the challenges of using our methods in protocols with natural language labels, and we show how the resulting method improves on the performance of using a well-known dictionary.

Summarizing, we present a vocabulary alignment method that is context-specific, lightweight, cheap and independent of external resources. This method can be used by agents as a low profile method of learning the vocabulary used in particular situations. We show that our method alone allows agents to find a useful alignment, although slowly. In combination with other resources, our technique provides not only a way of learning alignments faster, but also a way of obtaining different information (about the use of words in context) that may be difficult to find otherwise, and to repair external alignments.

Resum

Un dels objectius dels sistemes multiagent és permetre la col·laboració entre agents heterogenis. Això pot donar lloc a la interacció de participants amb diferents tipus de coneixements, habilitats i recursos, cosa que pot generar un entorn enriquidor. Perquè aquesta col·laboració es pugui realitzar, però, cal tenir en compte els diferents tipus d'heterogeneïtat que poden existir entre els agents. Un aspecte important és, per exemple, l'heterogeneïtat lingüística. Per poder coordinar les accions dels agents cal que puguin comunicar-se entre ells; i aquesta comunicació només pot reeixir si tots fan servir el mateix vocabulari i l'entenen de la mateixa manera.

El problema de l'entesa mútua entre agents amb diferents vocabularis ha estat, majoritàriament, analitzada amb tècniques que pressuposen l'existència d'elements externs comuns, tals com un meta-llenguatge, un entorn físic, o recursos semàntics. No obstant això, aquests elements no sempre estan disponibles. Fins i tot quan ho estan, és possible que generin alineaments que no siguin útils per a les interaccions que els agents volen dur a terme, atès que no estan contextualitzats.

Aquesta tesi proposa una visió diferent de l'alineament entre vocabularis, considerant agents que només comparteixen el coneixement sobre com dur a terme una tasca. Aquesta informació està especificada en un protocol d'interacció. Específicament, proposem la idea d'alineament basat en la interacció, en què els agents aprenen un alineament a força d'interactuar entre ells, observant allò que funciona i allò que no en llur conversa. Considerem, a tall d'exemple, la situació en què un turista intenta demanar un cafè en una llengua que no domina. Tot i que no hi ha un llenguatge comú, és probable que aquesta interacció acabi amb èxit, ja que està composta de passos simples en què tots coincideixen. Més encara, si la interacció es repeteix diverses vegades, és possible que el turista aprengui com es demana un cafè en l'idioma estranger. Tot i que aquest tipus d'adaptació resulta natural per als humans, aquesta idea encara no ha estat explorada en detall per a agents artificials.

Al llarg d'aquesta tesi estudiem com agents amb especificacions d'interaccions formalitzades de diferents maneres poden aprendre un vocabulari nou. Concretament, proposem tècniques d'alineament basades en la interacció per a protocols

especificats amb autòmats finits, amb restriccions lògiques, i amb semàntiques socials. Per a cadascun d'aquests casos provem tècniques que permeten inferir informació semàntica a partir d'interaccions o de l'observació d'interaccions entre altres agents. També analitzem com combinar aquestes tècniques amb alineaments externs, mostrant la manera en què es poden reparar quan contenen errors.

Els mètodes que proposem per a cada tipus d'especificació són avaluats mitjançant simulacions, usant protocols artificials generats aleatòriament. D'aquesta manera obtenim una avaluació general, que no està esbiaixada per les particularitats de les dades. A més, estudiem com aplicar els nostres mètodes a dades empíriques creades per humans, extretes del lloc web WikiHow. En aquesta avaluació discutim els desafiaments a què ens hem afrontat en aplicar els nostres mètodes al llenguatge natural, i mostrem que malgrat tot millorem els resultats obtinguts respecte a usar un diccionari.

En resum, en aquesta tesi proposem un mètode d'alineament de vocabularis que depèn del context i no requereix recursos externs, ni de la col·laboració d'altres agents. El nostre mètode, per si sol, permet trobar alineaments útils, però pot ser lent. Malgrat això, quan es combinen amb altres recursos, les nostres tècniques permeten agilitzar l'aprenentatge i reparar alineaments externs, alhora que proveeixen informació sobre l'ús de paraules en el seu context, quelcom difícil d'obtenir d'una altra manera.

Resumen

Uno de los objetivos de los sistemas multiagente es permitir la colaboración entre agentes heterogéneos. Esto puede resultar en la interacción entre participantes con distintos tipos de conocimiento, habilidades, y recursos, creando un ambiente abierto y diverso. Para que esta colaboración funcione, es necesario tener en cuenta los diferentes tipos de heterogeneidad que pueden existir entre los agentes; por ejemplo, la heterogeneidad lingüística. Para poder coordinar sus acciones, es necesario que los agentes puedan comunicarse entre ellos; y esta comunicación sólo puede ser exitosa si todos usan el mismo vocabulario, y lo entienden de la misma manera.

El problema del entendimiento mutuo entre agentes con diferentes vocabularios ha sido, mayoritariamente, analizado con técnicas que asumen la existencia de elementos externos comunes, como un meta-lenguaje, un ambiente físico, o recursos semánticos. Sin embargo, estos elementos no siempre están disponibles. Incluso cuando lo están, es posible que generen alineamientos que no sean útiles, en particular, para las interacciones que los agentes quieren completar, dado que no están contextualizados.

Esta tesis propone una visión diferente del alineamiento entre vocabularios, considerando agentes que solamente comparten el conocimiento sobre como llevar a cabo una tarea. Esta información está especificada en un protocolo de interacción. Específicamente, proponemos la idea de *alineamiento basado en la interacción*, en la cual los agentes aprenden un alineamiento a base de interactuar entre ellos, observando lo que funciona y lo que no en una conversación. La situación en la cual un turista intenta pedir un café en un idioma que no domina es una analogía útil. Aún cuando no hay un lenguaje común, es probable que esta interacción termine exitosamente, dado que está compuesta de pasos simples en los cuales todos coinciden. Más aún, si la interacción se repite varias veces, es posible que el turista aprenda como se pide café en el idioma extranjero. A pesar de que este tipo de adaptación resulta natural para humanos, esta idea aún no ha sido explorada en detalle para agentes artificial.

A lo largo de esta tesis estudiamos como agentes que tienen especificaciones formalizadas de diferentes maneras pueden aprender un vocabulario nuevo. Concretamente, proponemos técnicas de alineamiento basadas en la interacción

para protocolos especificados con autómatas, con restricciones lógicas, y con semánticas sociales. Para cada uno de estos casos, proveemos técnicas que permiten inferir información semántica a partir de interacciones, o de la observación de interacciones entre otros. También analizamos como combinar estas técnicas con alineamientos externos, mostrando como pueden repararlos cuando contienen errores.

Los métodos que proponemos para cada tipo de especificación son evaluados mediante simulaciones, usando protocolos artificiales generados aleatoriamente. De esta manera obtenemos una evaluación general, que no está sesgada por particularidades de los datos. Además, estudiamos como aplicar nuestros métodos a datos empíricos creados por humanos, extraídos de la página web *WikiHow*. En esta evaluación discutimos los desafíos enfrentados al aplicar nuestros métodos al lenguaje natural, y mostramos que mejoramos los resultados obtenidos al usar un reconocido diccionario.

En resumen, en esta tesis proponemos un método de alineamiento de vocabulario que depende del contexto y no requiere recursos externos, ni de la colaboración de otros agentes. Nuestro método, por si solo, permite encontar alineamientos útiles, pero puede ser lento. Sin embargo, cuando son combinadas con otros recursos, nuestras técnicas permiten agilizar el aprendizaje y reparar alineamientos externos, a la vez que proveen información sobre el uso de palabras en contexto, la cual puede ser difícil de obtener de otra manera.

Acknowledgments

I'm finally finishing my thesis. At many points during the three and a half years that I spent working on it I felt very identificated with that Talking Heads song that goes "and you may ask yourself, well, how did I get here?." Such an immersive task it was. It may be a bit early to think in retrospective, but I am sure it was an incredibly enriching experience on many different levels. There will be time to make a detailed personal balance, but one thing is clear: that a lot of people has been part of this. I am happy to have this opportunity to thank them for their help and company.

First of all, I want to thank my supervisor Marco Schorlemmer for his guidance and support during these years. Looking back, I realize his supervision has been a remarkable equilibrium between giving me space and freedom to explore new ideas, and helping me get back on track when I needed so. Thanks for the dedication and trust. I really appreciate it.

I want to thank my thesis committee: Jérôme Euzenat, Fiona McNeill and Enric Plaza. In addition to agreeing to assess my thesis, Jérôme has, during these years, taken the time to read my papers and provide interesting comments and suggestions. I enjoyed discussions with Fiona while I was in Edinburgh; and though I did not stay for long, I appreciate her insightful and supportive comments. Thanks as well to Dagmar Gromann for accepting to be my *expert reporter*, as well as for including me in the many activities she takes part in.

During the last year I had the opportunity to collaborate with Paolo Pareti, developing what is now one of the chapters of this thesis. This has been a great experience; it was exciting to see how our projects could be combined and enriched, and our discussions were always interesting. Thank you for sharing your data and your time. I hope we will work together again in the future.

I want to thank the people at the DISI department in the University of Trento, where I spent three months in 2015, for their warm welcoming. Mercedes, in particular, was definitely the best partner of hikes and lunches. I also want to mention Michael Rovatsos, who read my work and discussed it with me, in addition to bringing together many of the people I am mentioning here through the ESSENCE network.

When I started working at the IIIA everyone told me it was like a big family,

and I can understand why. It has been a pleasure to work here. The PhD's office has changed quite a bit since I arrived, so there are many people I would want to thank. Ewa, Kemo, Xavi, Marc, Jesús and Tito are some of the ones with whom I shared more time. También quiero agradecer a Ana Beltrán por su invaluable ayuda con asuntos burocráticos.

Barcelona is a wonderful city, but it wouldn't have been the same without the company of many friends. Algunos estuvieron ahi desde el principio, como Ceci, o casi, como Julia, mi eterna compañera de aventuras. In the last years I found an unlikely group of friends in the institute next door and now I know a lot more about Materials Science. Júlia, Laia, Alex, Sergio and Vish have been an amazing team for everything; from weekend excursions to *Bang!* games to holding a monthly seminar about random topics. I cannot finish this without mentioning Juan Carlos, the only person with whom I can walk slowly.

I was lucky to find a home that will be hard to leave. Thanks to Claudio, Sílvia, and Ana for building Tordera each day. A Claudio por sus palabras de aliento, a Sílvia por su energia infinita. A Ana, bueno, ya es difícil encontrar una compañera de piso que además sea una gran amiga. Si a eso le sumas que escriba su tesis justo al mismo tiempo... perfect match? Qué bueno que sobrevivimos. Te voy a extrañar.

Leaving my dear Buenos Aires was not easy, and I think I missed it a lot more than I thought I would. I want to thank my family in Buenos Aires: my mom and dad, to whom I owe so much, and my sister Alina. Gracias por todo, y perdón por no estar ahí. To my highschool friends, also known as *la primé*, les quiero agradecer que cuando vuelvo es como si no me hubiese ido nunca. I want to thank the University of Buenos Aires and everyone who contributes to its quality, because while writing this dissertation I realized surprisingly often how important that education was.

The last paragraph is for Thomas. He has helped this thesis directly in so many ways; from the beginning, discussing problems, to the last weeks of proofreading. And yet of course, that is a minimal fraction of what I have to thank him for. It is difficult to express how happy I am to have found such a great teammate, so I will just stop here. Gracias! Todo es mejor con vos.

Contents

Abstract **v**

Resum **vii**

Resumen **ix**

Acknowledgments **xi**

1 Introduction **1**
1.1 Interaction-based Vocabulary Alignment 3
1.1.1 For Humans 5
1.1.2 For Artificial Agents 6
1.2 Interaction Protocol Specifications 7
1.3 Contributions and Organization 10
1.4 Publications 13

2 Computational Approaches **15**
2.1 Vocabulary Alignment in Artificial Environments 15
2.2 Meaning as Relations between Words 17
2.3 Meaning as Relations between Words and Objects 21
2.4 Meaning as Relations between Words and Effects in an Interaction 24
2.5 Discussion 26
2.5.1 What Needs to be Shared 27
2.5.2 Combining Sources of Meaning 28
2.5.3 Our Contribution in Context 29

3 Integrating Alignment Techniques **31**
3.1 Introduction 31
3.2 Performing Tasks by Interacting 34
3.3 Learning from the Experience of Interaction 36
3.3.1 Interaction Models 37
3.3.2 Interaction Dynamics 41

3.3.3 Alignment Technique . . . 43
3.4 Using External Vocabulary Alignments . . . 45
3.5 A Combined Technique . . . 49
3.5.1 Learning from Unsuccessful Experiences . . . 50
3.5.2 The Combination Methods in Action . . . 53
3.6 Experimental Evaluation . . . 53
3.6.1 Data Generation . . . 53
3.6.2 Experiments . . . 54
3.7 A Pragmatic Approach to Alignment Evaluation . . . 58
3.7.1 Pragmatic Precision and Recall . . . 59
3.7.2 Pragmatic Precision and Recall in Practice . . . 62
3.8 Conclusion . . . 64

4 Open Protocols **67**
4.1 Introduction . . . 67
4.2 Open Interaction Protocols . . . 69
4.2.1 Preliminaries . . . 69
4.2.2 Open Protocols as Interaction Protocols . . . 71
4.3 Communicating with Heterogeneous Partners . . . 77
4.4 Learning Translations from Interactions . . . 80
4.4.1 Estimating the Confidence in the Interaction . . . 83
4.4.2 Students and Teachers: Agents that Actively Want to Learn 86
4.5 Experimental Evaluation . . . 87
4.6 Including p-Necessary Constraints . . . 93
4.7 A Purely Logical Approach to Alignment Inference from Interactions . . . 95
4.8 Vocabulary Alignment for Agents with Flexible Protocols . . . 98
4.8.1 Updating Technique . . . 100
4.8.2 Experimental Evaluation . . . 101
4.8.3 Discussion . . . 104
4.9 Conclusions . . . 105

5 Inference of Commitment Semantics **107**
5.1 Introduction . . . 107
5.2 Scenarios . . . 109
5.3 Commitment Specifications . . . 111
5.3.1 Syntax . . . 112
5.3.2 Semantics . . . 113
5.4 Inferring Social Semantics . . . 115
5.5 Performance Analysis . . . 118
5.5.1 Experiment 1 . . . 119
5.5.2 Experiment 2 . . . 119

5.6 Semantic Extensions . . . 121
5.6.1 Frequency . . . 121
5.6.2 Observing Punishments . . . 122
5.6.3 Cancelation Policies . . . 123
5.7 Conclusions . . . 125

6 A Study with Empirical Data **127**
6.1 Introduction . . . 128
6.2 A Model of Protocols . . . 131
6.3 Performing Tasks Collaboratively . . . 132
6.3.1 Collaboration Dynamics . . . 133
6.4 Translation Learning Technique . . . 135
6.4.1 Choosing a Mapping . . . 136
6.4.2 Updating ω . . . 138
6.5 Data Acquisition . . . 139
6.5.1 Instruction Selection . . . 139
6.5.2 Conversion of Instructions into Protocols . . . 140
6.5.3 Protocol Selection . . . 140
6.5.4 Label Cleaning . . . 141
6.5.5 Identification of the Semantically Correct Translation . . . 141
6.6 Evaluation . . . 142
6.6.1 Success Rate . . . 142
6.6.2 Translation Quality . . . 144
6.6.3 Evaluation Discussion . . . 149
6.7 Conclusions . . . 149

7 Conclusion **151**
7.1 Differences between Protocol Specifications . . . 153
7.2 Combination with Other Sources of Meaning . . . 154
7.3 Non-Compatible Protocols . . . 155
7.4 Relation with Grammar and Language Structure . . . 156

Bibliography **157**

A References to Software **173**
A.1 Software for Chapter 3 . . . 173
A.2 Software for Chapter 4 . . . 174
A.3 Software for Chapter 5 . . . 174

List of Figures

1.1 Transition-based specification to order drinks 9
1.2 Constraint-based specification to order drinks 10

2.1 A simplified OWL ontology 18
2.2 A simplified taxonomy 19
2.3 A possible observed situation for cross-situational learning 23
2.4 Transition-based *ordering drinks* protocol 25
2.5 I-SSA alignment protocol 26

3.1 Interaction Models for the *travel agency* scenario 39
3.2 Results for experiment 1 56
3.3 Results for experiment 2 57
3.4 Interaction models for the *ordering drinks* scenario 61
3.5 Success rates for different values of ξ' 63

4.1 F-score curve for different agents 89
4.2 Results for different protocol sizes (experiment 1) 90
4.3 Results for agents with previous alignments (experiment 2) . . . 91
4.4 Results for different word distributions (experiment 3) 92
4.5 Results for one-side learning (experiment 4) 93
4.6 Results for minimizing punishment in flexible protocols 102
4.7 Results for flexible protocols with variations 103

5.1 Convergence for different specification types 121
5.2 Frequency 122
5.3 Punishments and policy 124

6.1 Success rate for different number of training interactions. 143
6.2 Precision against the Oxford Dictionary reference 145

List of Tables

3.1 Extract of an alignment obtained with an ontology matcher for the *travel agency* scenario 48
3.2 Results for the *travel agency* scenario 53
3.3 Two alignments for the *ordering drinks* example 62

4.1 LTL definitions of constraints 72
4.2 Updates for each violated constraint (agent reasoner) 85
4.3 Average number of interactions before convergence 95
4.4 Updates for each violated constraint. 97
4.5 Convergence for vocabularies of size 8 97

5.1 Proportion of compliant interactions for different numbers of observed interactions 120

6.1 Translation sizes after 2048 training interactions 146
6.2 Expert evaluation 147
6.3 Crowdsourcing evaluation 148

Chapter 1

Introduction

> "I speak and speak," Marco says, "but the listener retains only the words he is expecting. The description of the world to which you lend a benevolent ear is one thing; the description that will go the rounds of the groups of stevedores and gondoliers on the street outside my house the day of my return is another; and yet another, that which I might dictate late in life, if I were taken prisoner by Genoese pirates and put in irons in the same cell with a writer of adventure stories. It is not the voice that commands the story: it is the ear."
>
> Italo Calvino, *The Invisible Cities*

The relationship between autonomous individuals and collaborative communities lies at the heart of multiagent systems. Consider some of the scenarios that may soon be ubiquitous, such as autonomous cars that need to inform each other of their position to avoid accidents and make traffic fluid; or medical assistant robots who follow orders from humans in hospitals; or even argumentative agents that defend different positions in a debate to solve a problem in a way that is fair for various stakeholders. Collaboration between diverse individual agents (which can have different manufacturers, be human or not, or have different interests) to achieve a common goal constitutes the richness of multiagent systems.

As evidenced in the previous examples, some kind of communication is necessary to bridge the gap between autonomous individuals and collaborative societies. By interacting, agents can exchange information about their situation, preferences, or available resources. Obtaining this information is essential to adapt to different situations and interlocutors in any truly collaborative task. The type of required communication depends on the abilities of the agents and

the complexity of the tasks to be performed. It can range from a binary signal in very simple scenarios to the different existing artificial languages, and to natural languages in situations where humans are also involved.

Of course, to ensure meaningful communication it is essential that the vocabulary that is used, together with its semantics, is shared by all interlocutors. A simple way to guarantee that all agents understand words in the same way is by establishing a central language, that is, the specification of a vocabulary and its semantics that everyone is aware of. However, the implementation of such a central specification is not straightforward. Multiagent systems are conceived as open environments where agents with different backgrounds interact with previously unknown partners. In such a dynamic, distributed environment, a first question consists in who defines the semantics that is going to be used. Different communities with particular needs could prefer to use different types of languages, so defining a central one could be a source of conflict. Moreover, if a community already has is own language, it will likely be reluctant to change it, since this can be costly. In this sense, semantic specifications correspond to the idea of *convention* proposed by Lewis (1969): something that no one in a community wishes to change as long as everyone else continues following it. The problem of vocabulary heterogeneity becomes even more evident if one considers that, increasingly, artificial agents are expected to collaborate not only between them but also with humans. It is unrealistic to expect all humans to speak the same language, even when agents do.

A possible solution is to let each community have a local language, but to make an ontology or semantic specification openly available so that foreign agents can learn it before starting to interact. This may not always possible either, for different reasons. First, the meaning of words that speakers use without any problem can be difficult to define precisely in an explicit, formal way. Even when this can be done, language is not static, and the meaning of words is in constant evolution, which could make specifications be soon deprecated. To illustrate these two complexities, it can be useful to compare everyday slang with the language described in a dictionary. The English word *hip*, for example, has evolved in the 20th century from meaning *someone or something into the jazz culture* to mean *beatnik*, *hippie*, and finally *hipster*. Its exact meaning is difficult to describe, but English speakers, at least of a particular age and social range, know what the word refers to and can use it without trouble (Leland 2009). Making specifications available raises another, more concrete, problem. In order for foreign agents to understand the specification of the semantics, it needs to be written in a common language that everyone can read. This solution, therefore, only takes the problem one level of abstraction higher, since it requires a shared meta-language to define specifications. Finally, this apparent solution may also be a source of conflict, since communities could be unwilling to share their entire ontology with any foreign agent for privacy reasons.

The situation we described exposes the need for a dynamic approach to language alignment. The multiagent community has proposed different solutions to this problem, that we will review in detail in Chapter 2. In broad terms, these contributions frame the problem from two different perspectives. Some approaches (for example the work by Steels (1998) and by van Diggelen et al. (2007)) consider the existence of external *contextual* elements that all agents perceive in common, and explore how these elements can be used to align the meaning of words. A second group of techniques, such as the ones proposed by Santos et al. (2016a) and by Silva et al. (2005) consider a situation in which this kind of context is not available, for example because agents communicate remotely or lack environmental sensors. To this end, they provide explicit ways of learning or agreeing on a common vocabulary (or alignment between vocabularies). These techniques require agents to share a common meta-language that they can use to discuss about the meaning of words and their alignments.

The complex question of how to communicate with heterogeneous interlocutors when neither a physical context nor a meta-language are available remains practically unexplored. In this dissertation we focus on alternative vocabulary learning methods that consider this situation. Instead of relying on external elements, agents infer an alignment from the information they have about the interactions they are taking part in. We assume that the procedural information about tasks that are performed –the interaction protocol– is shared by all interlocutors, and we study how agents can infer alignments only from this information. They do so by interacting repeatedly, observing the outcomes of different utterances. In short, the question we address in the following chapters can be summarized as:

Can artificial agents learn vocabularies from repeated interaction, when they only share structural information about the specification of the tasks they perform?

We will refer to this idea, explained in detail in what follows, as *interaction-based vocabulary alignment*.

1.1 Interaction-based Vocabulary Alignment

Let us introduce the idea with a simple example. Suppose an English speaker is trying to buy a coffee in Italy, in a bar where the waiter only speaks Italian. Even if they do not speak each other's languages, the interaction is likely to be successful. This is, in part, thanks to two factors. First, the customer and the waiter (mostly) agree on how the interaction for buying coffee should unfold. That is, the customer has an idea of how she would perform the interaction with an English waiter; and this interaction, translated to Italian, corresponds

to the Italian waiter's idea of how to sell coffee. Of course, there can be cultural variations between how coffee is served in different parts of the world, but we do not go into such details. Second, the interaction context is very reduced; the customer is not expecting the waiter to talk about what she had for lunch or about politics. This implies that the number of possible mappings can be kept small, and therefore easier to discover. This can be summarized by two observations:

1. Understanding a foreign language is easier when the context is familiar.
2. Understanding a foreign language is easier when the context is restricted.

Even if the interlocutors fail to perform the task correctly the first time they try, it is likely that, if they continue performing it, they will eventually become able to succeed. In particular, the English customer will learn how to understand the task in Italian after some experiences, observing which utterances work and which ones do not. Simple and frequent tasks are rapidly mastered by foreign speakers, since they have plenty of opportunities to practice. The fact that individuals learn from repeated evidence has been extensively studied in animal psychology (Thorndike 1898, Herrnstein 1970) and is at the core of machine learning approaches such as reinforcement learning (Sutton and Barto 1998). This corresponds to a third observation:

3. Understanding a foreign language is easier when a situation occurs repeatedly, assuming agents can learn from these interactions.

It is important, at this point, to clarify what we mean with *understanding*. In our example, the objective of the interlocutors is to succeed at one particular task: buying or selling coffee. Indeed, the language learning scenario that we propose provides automatically a notion of successful language understanding: the success of the task that is being performed. The success of the task, meanwhile, is defined in a procedural specification; the execution of a task succeeds if it follows the specification, and fails if it does not. Summarizing, we say an agent understands a language if it can use it in a way that complies with a specification. For example, we would say a customer understands the question *what do you want to drink?* if she answers *coffee*, and does not understand it if she answers *a cheese sandwich*.[1]

Taking into account the three observations, in this dissertation we propose computational techniques to learn foreign vocabularies by interacting repeatedly

[1]The idea of considering knowledge as successful behavior is described in Parikh (1994) with a useful example. Consider a mouse that is put in a maze where some cheese is hidden always in the same place. If the mouse always manages to find the cheese, we say it *knows* where it is. Even if we do not actually know how it obtains the information, observing that it succeeds is enough to assume knowledge.

in situations that are well-known to all interlocutors. In a way, our agents follow the same method as Quine's radical translator to build a manual:

> "Our radical translator would put his developing manual of translation continually to use, and go on revising it in the light of his successes and failures of communication. And wherein do these successes and failures consist, or how are they to be recognized? Successful negotiation with natives is taken as evidence that the manual is progressing well. Smooth conversation is further favorable evidence. Reactions of astonishment or bewilderment on a native's part, or seemingly irrelevant responses, tend to suggest that the manual has gone wrong." (Quine 1987, p.7)

The idea of learning from interacting and observing outcomes has been extensively developed for humans, but not so much for artificial agents. Let us discuss these two cases in the following subsections.

1.1.1 For Humans

The text by Quine that we quote above discusses philosophically the alignment of language in context, from the experience of interacting. This idea has also been investigated from a cognitive point of view, as well as in practical applications to second language acquisition. In the first case, cognitive studies have shown that human pairs performing a task together align the meaning of words they use to make the communication simpler or more efficient. For example, Pickering and Garrod (2004) discuss how pairs of interlocutors align their meaning representations while performing a concrete task, such as describing their positions in a maze. The article points out that interlocutors align the way in which they speak during the dialog; for example, they implicitly decide on task-specific meanings for particular expressions and continue using them in the convened way, or they imitate particular vocabulary choices their interlocutors make. Linguistic alignment is a famous and pervasive phenomenon that has been found to take place in many levels, such as phonetics, syntax, and vocabulary, as described in (Pickering and Garrod 2004).

Among second-language acquisition approaches, the idea of learning vocabulary from interacting in concrete tasks is an increasingly popular one. This idea is in general known as *Task-Based Language Teaching* (Nunan 2006), and it proposes the use of everyday interactive situations to practice a foreign language. The tasks that are used have a clear non-linguistic objective, that students need to achieve using their available linguistic resources. Originally conceived as a tool to be included in the classroom, this idea has profited from new technologies, using for example social networks (Mills 2011) or augmented reality (Santos et al. 2016b) to create interactive situations.

1.1.2 For Artificial Agents

Software agents are usually designed to perform concrete tasks, and have access to a large number of them, either because they perform them themselves or because they can access a database. For this reason it seems natural to apply a technique based on observing repeated interactions to the problem or artificial language alignment. However, the ideas we have just explained, although extensively investigated for humans, are relatively unexplored for the case of artificial agents that need to align their vocabularies. One of the first attempts to formalize an interaction-based vocabulary alignment technique for artificial agents is the work in Atencia and Schorlemmer (2012). We will take this work, that we explain in detail in Chapter 2, as a basis for our investigation.

We consider two interacting agents that perform a task together following a specification. These interactions are performed by sending messages to each other, but our agents speak different languages, so they cannot interoperate directly. If an agent sends messages in its own language, the other one would not know how to interpret them, and would therefore be unable to continue the interaction. We assume that interlocutors do share the knowledge of how the interaction is performed. That is, the structural part of their specifications is similar. Continuing with the coffee example, both the customer and the waiter would agree that coffee can be ordered when a customer is asked what she wants to drink, (even when one agent says *coffee* and the other one *caffe*) but a cheese sandwich cannot be ordered after this query (even when one agent would say *cheese sandwich* and the other one *panino al formaggio*). Put differently, there exists an alignment such that, if agents knew it, they could interact with each other without any problem.

Sharing the structure of the interaction specification means that, at least when agents interpret messages correctly, they have the same expectations about what can be said. The techniques proposed in Atencia and Schorlemmer (2012) that we take as a basis and develop in this thesis, propose to use this shared information to find an alignment between vocabularies. More concretely, agents learn mappings between their vocabularies by analyzing the messages they receive and comparing them with the messages that they were expecting, that is, the possible interpretations. For example, if the waiter asks what the customer wants to drink and receives *tea* as an answer, she will infer that *tea* does not mean *panino al formaggio* (cheese sandwich in Italian). Of course, this only makes sense if the customer understood the question in the first place. This uncertainty is what adds complexity to the problem.

An important advantage of the methods that we propose is they do not rely on any external resource. We only assume the existence of a shared idea about how to perform the tasks, which is necessary to interact. Moreover, this structure actually has to be shared, since otherwise the agents would not be

able to act together without previously aligning their specifications, even if they spoke the same language.

From a technical point of view, we will use an iterative and probabilistic approach to learning. In the techniques that we propose each agent has a *confidence distribution* that assigns a value to each mapping between a local and a foreign word. These values represent how confident the agent is on each mapping. We assume agents interact with each other repeatedly, updating the values in the confidence distribution with the experience gained in these interactions. A probabilistic approach is a natural way of dealing with the uncertainty that we discussed before, and has several advantages that we will discuss in depth in Chapter 4. We explain the updating techniques that we propose using well-known learning frameworks, which allows us to gain better understanding of the advantages and limitations of our methods, and to reuse existing solutions.

To keep the problem manageable, throughout this dissertation we make a number of simplifications that can be unlikely in real world scenarios. To start, we consider agents that learn an alignment only from the information in the specification, when they normally would use a combination of available resources, such as visual cues, syntactic reminiscence, etc. These, however, are additions that would simplify the problem. Our objective is to show that, under certain conditions, the interaction specification is enough to obtain an alignment that is good enough to interact meaningfully. Another simplification consists in assuming that agents share completely the knowledge about how to perform their tasks, when it is natural to have certain differences. This assumption is useful to study how an alignment can be extracted from shared procedural knowledge. Since we use probabilistic techniques, minor differences can be considered as noise that would not affect the learning process. We discuss how to deal with major differences at the end of Chapter 4.

1.2 Interaction Protocol Specifications

To define methods that find alignments from the execution of a common interaction protocol, it is necessary to formally define how this information is specified. Depending on this, different techniques will be useful to extract the information that is necessary to infer an alignment. The question of what should be expressed by an interaction protocol has been discussed extensively among the community working on multiagent communication in the 90s and 00s (Chopra et al. 2013). As a result, different techniques to formalize the *knowledge of how to perform a task* have been proposed. In this dissertation we explore alignment methods for different ways of specifying communication protocols. In what follows we briefly review different existent types of multiagent protocol specifications.

Agent Communication Languages

Agent Communication Languages (ACLs, Labrou et al. (1999)) result from efforts, conducted in the 90s, to achieve a standard formalization to describe interactions between artificial software agents. Two prominent examples are FIPA (Poslad et al. 2000) and KQML (Finin et al. 1994). These languages are inspired by the idea of Speech Acts (see Searle (1969) and our discussion in Chapter 3), which states that the utterance of messages perform actions in the real world. Concretely, ACLs consist of *performatives*, which are types of messages with particular semantics. For example, by uttering the performative *REQUEST*, an agent can ask another one to do something. Although different options have been proposed, the original semantics are based on mental states of the interlocutors; concretely using the BDI theory (Beliefs, Desires and Intentions, described by Rao and Georgeff (1995)). Larger protocols can be specified as sequences of performatives. This is, in general, not simple, since performatives have very specific meanings and therefore the flexibility is limited. Moreover, when based on mental states, the semantics of performatives are not verifiable externally; that is, it is not possible to decide if an agent has complied or not with a protocol by observing an interaction. For these reasons, ACLs are not often used in practice anymore.

Methods based on Transition Systems

Perhaps the most straightforward way of specifying interaction protocols is by using finite state machines (see, for example, the work by Barbuceanu and Fox (1995)). The way in which these formalizations represent the flow of an interaction is simple: in each state, any of the messages in the outgoing arrows can be uttered, and only those. By uttering messages, agents change states. An example of a finite state machine that specifies an interaction to order drinks is provided in Figure 1.1.

This simple idea has been extended in different ways to increase its expressive power. A limitation of finite state automata is that they cannot specify concurrent actions. This feature was included in different extensions. For example, Cost et al. (2000) propose using Petri Nets to specify interactions, a formalism that naturally incorporates support for concurrency. The models known as Communicating Finite State Machines (Brand and Zafiropulo 1983) are another approach that tackles this problem. These models are used, for example, in the communication formalism proposed for the Electronic Institutions framework (d'Inverno et al. 2012), which also incorporate constraints to the protocol, in the form of pre- and post-conditions to utterances that specify when messages can be uttered. Finally, the Lightweight Coordination Calculus (Robertson 2004) can also express concurrency. This language is based on a

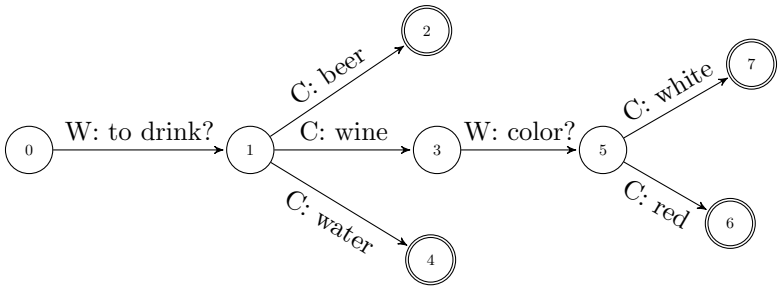


Figure 1.1: Transition-based specification to order drinks between a customer (C) and a waiter (W). Each arrow is labeled with a message, that includes the name of the sender and the utterance.

process calculus (such as Communicating Sequential Processes (Hoare 1985)), instead of on a finite state machine.

An advantage of transition systems is that they are very simple to verify. However, they have a central problem: their rigidity. These methods describe the complete possible flows of interaction, meaning that it is necessary to specify *all* the messages that can be possible at each state. Moreover, they fix a particular order for the executions, which is in many cases not necessary. While this may be easy to work around for short and very restricted interactions, in more complex cases it makes the protocols very difficult to design, read, change, and reuse.

Methods based on Social Norms and Rules

A way of solving the problem of protocol rigidity is to use a formalism that does not require the specification of the full interaction flow, but rather describes what can be said in more abstract terms. That is, to define *declarative* protocols, specifying what can be said, but not necessarily how. A simple way of doing this is by using a constraint language that describes rules over the utterance of messages. Examples are the approaches by Giordano et al. (2007) and by Montali (2010), which use Dynamic Linear Temporal Logic and Linear Temporal Logic respectively. These protocols determine rules about which utterances can be made before or after others. Notice that with this kind of specification, the complete order of messages can be determined if necessary, but this is not mandatory as with transition-based systems, thus avoiding over-specification. Figure 1.2 shows a specification of an interaction to order drinks based on constraints. While the transition-based protocol in Figure 1.1 defined everything that could be said after each state, this one only establishes some constraints that need to be followed. Any utterance that does not violate a constraint is

$$\{\mathrm{F}\ say(W, \mathsf{to\ drink}),$$
$$\mathrm{G}\,(\mathrm{X}\ say(C, \mathsf{beer}) \rightarrow say(W, \mathsf{to\ drink})),$$
$$\mathrm{G}\,(\mathrm{X}\ say(C, \mathsf{wine}) \rightarrow say(W, \mathsf{to\ drink})),$$
$$\mathrm{G}\,(say(C, \mathsf{wine}) \rightarrow \mathrm{F}\ say(W, \mathsf{color}))\}$$

Figure 1.2: Constraint-based specification to order drinks between a customer (C) and a waiter (W). Uses LTL logic, where F intuitively means *at some point in the future* and G means *at every point in the future*, and X means *in the next point in the future*. The constraints determine rules about who can say what and when.

possible.

A different approach to specifying protocols with social norms is the one known as *commitment semantics* (Singh 2000). Commitments are contracts that agents can enter between each other. For example, a waiter and a customer can establish a commitment stating that the waiter will bring coffee if the customer pays. In a way, this may seem equivalent to having a rule establishing this mandatory behavior. However, commitment specifications go one step further by allowing agents to operate with commitments, creating and canceling them. Instead of being external, rules are something agents can decide about and manipulate. These specifications, although very flexible, may not suffice to express more regulative aspects of interactions, such as temporal constraints. For this reason, approaches that combine commitments with information about the flow of the interaction have been proposed (Baldoni et al. 2013). Formally, an example of a commitment specification looks like:

`Accept order` **means** Create_commitment(`Pay`, `Coffee`)

That is, by accepting an order, a waiter is committing to bring coffee if the customer pays.

1.3 Contributions and Organization

In this dissertation we study how the idea of interaction-based vocabulary alignment can be applied to different situations and, particularly, to different types of interaction specifications, analyzing the complexity that each formalization presents. We use as a basis the ideas in Atencia and Schorlemmer (2012) (that we explain in detail in Chapter 2), where a simple alignment technique is proposed for protocols based on transitions. In the following chapters we extend

these ideas, considering more complex scenarios, as well as more sophisticated techniques. The central extensions can be summarized in four points:

- We consider different, and more realistic, specification protocols.
- We define alignment techniques in the context of well-know learning frameworks.
- We consider the interaction of these techniques with external alignments.
- We discuss different learning strategies.
- We apply the ideas to real data.

The first three points are developed in Chapters 3, 4, and 5, and the last one in Chapter 6. The chapters in this dissertation are organized to explore different types of protocol specifications. For each case we develop alignment techniques with different extensions, and evaluate them experimentally, using code that is publicly available. Concretely, we study simple cases of transition- and constraint-based specifications, as well as those based on commitments. In the final chapter, we present an application to real data. Let us now describe in more detail the content of each chapter, and the publications on which they are based.

Chapter 2 As we have already mentioned, the problem of semantic heterogeneity in multiagent systems has been approached from different points of view. In this chapter we present a review of existing computational techniques to align vocabularies. We classify the methods into three classes, based on which type of meaning representation they use. The first class considers meaning to be represented as relations between words, usually specified in ontologies. The second one relates words with objects. The third one relates words with the effect their utterance has in the context of an interaction. We review the most prominent methods in these categories, and discuss how they relate to the approach in this dissertation.

Chapter 3 The objective of this chapter is twofold. First, it presents our extended version of the methods proposed in Atencia and Schorlemmer (2012). Concretely, we formulate the idea of interaction-based vocabulary alignment as a reinforcement learning technique. Second, we discuss how this technique can be used together with external alignments. The new formulation as a learning technique allows us to propose, in a simple way, different methods to take previous knowledge into account. We also present the basics of an evaluation setting for external alignments that uses a multiagent interaction, instead of a human-crafted gold standard. This chapter is based on the work in (Chocron and Schorlemmer 2016a) and in (Chocron and Schorlemmer 2016c).

Chapter 4 In this chapter we reformulate the ideas in Chapter 3 to apply them to the constraint-based specifications that we described in Section 1.2. Since these are much less informative, the learning strategy needs to be cross-situational; that is, agents learn from interacting in many different situations. We discuss different ways of inferring alignments from tasks, and perform experiments for different situations. Finally, we discuss how to relax the assumption of agents having a shared knowledge of how to perform tasks, considering how they can learn translations that let them interact even when they have different interests. This chapter is based on the work in (Chocron and Schorlemmer 2017a) and in (Chocron and Schorlemmer 2017b) .

Chapter 5 In this chapter we move on to consider a third way of representing interactions, the previously mentioned *commitment semantics*. Here, we consider a slightly different problem. Instead of assuming two agents that need to align the words they use, we focus on how an agent can learn, from scratch, the semantics of a previously unknown vocabulary, by observing other agents interacting. We discuss under which conditions this is possible, and evaluate the methods on randomly generated data. This chapter is based on the work in (Chocron and Schorlemmer 2018).

Chapter 6 Until now, all the methods we proposed were evaluated on randomly generated data. In this chapter we study an application of our techniques to real-world protocols. Concretely, we use protocols from the WikiHow website, which establish how to do something in different languages. These protocols are built by humans, and written in natural language. We use a cross-situational technique as in Chapter 4 to get correct alignments. This is necessary because the protocols are very simple, so performing one is not enough to learn an alignment. We explore the challenges that arise when considering natural language protocol labels; mainly, the fact that labels are sentences with multiple words.

Chapter 7 This chapter is the conclusion of the dissertation, where we present a discussion of the obtained result in a transversal way that considers the different techniques that we proposed. We also discuss the directions of research that this dissertation opens.

Appendix A The software that we used to evaluate the techniques that we propose in this dissertation is publicly available in online repositories to make experiments reproducible and allow for the reuse of the code. In this appendix we provide links to the corresponding repositories and comment on what can be found in each of them.

1.4 Publications

We summarize here the publications that have been derived from this dissertation:

- Paula Chocron, Marco Schorlemmer; Inferring Commitment Semantics in Multi-Agent Interactions, 2018. Accepted for publication at the International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS).

- Paula Chocron, Marco Schorlemmer; Vocabulary Alignment for Agents with Flexible Protocols, 2017. Proceedings of the Joint Ontology Workshops (JOWO).

- Paula Chocron, Marco Schorlemmer; Vocabulary Alignment in Openly Specified Interactions, 2017. Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems, (AAMAS).

- Paula Chocron, Marco Schorlemmer; Attuning Ontology Alignments to Semantically Heterogeneous Multi-Agent Interactions, 2016. Proceedings of the 22nd European Conference on Artificial Intelligence (ECAI).

- Paula Chocron, Marco Schorlemmer; Interaction Specifications as Contexts for Ontologies, 2016. Proceedings of the Joint Ontology Workshops, co-located with FOIS (JOWO).

- Paula Chocron, Marco Schorlemmer; Ontology Alignment Evaluation in the Context of Multi-Agent Interactions, 2016. Proceedings of the 11th International Workshop on Ontology Matching, co-located with the 15th International Semantic Web Conference (ISWC).

- Paula Chocron, Marco Schorlemmer; Social Coordination Systems with Ontology and Protocol Heterogeneity, 2015. Artificial Intelligence Research and Development. Proceedings of the 18th International Conference of the Catalan Association for Artificial Intelligence (CCIA).

Chapter 2

Computational Approaches to Vocabulary Alignment

> All language is a set of symbols whose use among its speakers assumes a shared past.
>
> Jorge Luis Borges, *The Aleph*

In Chapter 1 we introduced the core problem addressed in this dissertation, that can be summarized as that of vocabulary alignment between agents that need to interact together and only share information about how tasks are performed. In this chapter we will discuss existent work that tackles similar questions. Concretely, we review different solutions to the problem of finding an alignment, organizing them according to the semantic resource that they use.

2.1 Vocabulary Alignment in Artificial Environments

The general question that underlies the research in this dissertation is that of how interlocutors can agree on the meaning of the words they use. This problem has been extensively discussed and it lies at the core of the question of how individuals interact and coordinate with each other in societies. From a philosophical point of view, different authors have discussed, first, whether the semantics of natural languages is actually shared by speakers and, if it is, how this is achieved (Lewis 1969, Quine 1960, Wittgenstein 1953).

In this chapter we focus on computational approaches to the problem of how two interlocutors can reach a shared semantics. We use the term computational to describe a technique that can be implemented and empirically evaluated, as opposed to the general philosophical terms in which the problem is often discussed. Concretely, the methods we review in this chapter allow artificial

agents to understand each other, even if they originally did not use the same vocabulary.

Crafting agents that can autonomously learn how to speak has been one of the objectives of the community working on multiagent systems for decades (Carlson et al. 2010, Cassell 2000). Languages are complex and evolving systems, and the problem of natural language understanding is considered to be an AI-complete problem (Ide and Véronis 1998). That is, a problem such that solving it "is equivalent to solving the entire AI problem - producing a generally intelligent computer program" (Shapiro 1992, Second Edition, p.54). In this chapter we review different approaches that consider the situation of an artificial agent that needs to interact with another one, who uses a foreign vocabulary, to perform some task collaboratively. The solutions that we discuss provide techniques that the agent can use to find, in an autonomous way, a translation of foreign words into its own vocabulary, allowing it to communicate meaningfully.

To describe how a translation between foreign and known words can be discovered, it is first necessary to determine when two words are equivalent. When is a word a translation of another one? This is closely related to the question of what meaning is and how it is represented. This issue has also been deeply discussed from a philosophical perspective; however, in order to work with meaning computationally, it is necessary to have a concrete definition of semantics. The development and evaluation of techniques to discover the meaning of words are only possible with a formal notion of meaning. In this section we discuss semantic alignment techniques that use different ways of specifying meaning.

Concretely, we review techniques that propose solutions to the problem of semantic heterogeneity in multiagent systems considering three different dimensions of meaning representation. First, we consider meaning to be relations between words themselves. This is a traditional approach to express semantics computationally, largely developed in the field of Knowledge Representation (Sowa 2000). We review existent methods to map words according to this relational information and show how they can be applied in a multiagent environment. Second, we consider meaning to be a relation between words and real word entities. This approach, in which each word represents a physical object, is close to the *referential* views of language, which discuss how names relate with things that they refer to (see (Frege 1948), (Russell 1905), and (Kripke 1977) for different versions of this approach). Third, we discuss a pragmatic approach, that considers meaning to be determined by how words are used in an interaction and, more concretely, by the effects their utterance has. Unlike the two previous approaches, this view considers words as part of a communicative process, analyzing their social effect. Meaning is not an isolated phenomenon, but something that occurs when different individuals interact. Vocabulary alignment within this view has been less explored, and it is the approach used in this

dissertation.

These three representations are abstractions and it is not implied that they encode, by themselves, the meaning of words in all the rich complexity they show in natural environments. We consider them to be different dimensions of meaning that can be useful for different applications. We focus on alignments obtained from each dimension separately because it keeps the approaches simple and minimizes the assumptions about what information is available. In the last section of this chapter we discuss, among other things, possible combinations between the methods we presented. As a last remark, let us mention that there exist efficient machine translation technologies that can be used to find translations between words in different languages (Goldberg and Levy 2014). However, they usually depend on the existence of a large corpus of data, and therefore only work for well-documented natural languages with a rich historical record, and for well-established ways of speaking within these languages. More importantly, they do not provide any insight on the problem of learning new vocabularies, something that is obtained with the methods that we propose here.

2.2 Meaning as Relations between Words

Let us start by considering a specification of meaning given by the relations that hold between words in a vocabulary. In analogy with human vocabulary learning, this corresponds to a learner asking for the meaning of, for example, the word *rabbit* and receiving as answer that it is a *small mammal*. In information science, the structures that organize information about relations between concepts are usually known as *ontologies*.

There exist many different languages to specify ontologies, each useful for different purposes. Perhaps the ones that are most widely used in computational environments, such as the Ontology Web Language (known as OWL, McGuinness et al. (2004)), are based on Description Logic, a formalism specifically designed to represent knowledge (see Baader et al. 2003 for a very comprehensive introduction). This is thanks to its convenient computational properties: description logic is, in general, decidable, and there exist efficient decision procedures for it. This logic can express two different kinds of relations. On one side, it can describe relations between concepts, such as, for example, *Coffee Shop is more general than Shop*. On the other side, it can express relations between individuals and concepts, such as *Sandwichez is a Coffee Shop*. An example of a (simplified) OWL ontology for the domain of coffee shops is provided in Figure 2.1. A different type of knowledge representation technique, known as *taxonomies*, use a much less descriptive formalism. Taxonomies are a tree structure that represent a hierarchical classification, where concepts are associated

by *is-a* relations. Figure 2.2 shows relations between drinks in a taxonomy.

As we discussed in Chapter 1, in open environments where agents with different backgrounds interact, it can be difficult to establish a central ontology that is useful and known to everyone. It is reasonable for each community to use the ontologies that are more adequate for its needs and resources, which may not be the same for others. For example, the ontologies we present in this section can be useful for a customer wanting to get a coffee, but a chemical laboratory could prefer to classify coffee as a `Liquid`. Once a structure has been chosen, its users are usually reluctant to change it, since the costs of the transition can be very high. This introduces the problem of semantic interoperability (Heiler 1995): processes that need to share information may have different names for the same concept or interpret the same name differently. To address this issue, a variety of techniques to find correspondences between concepts in different ontologies have been developed. A good survey of these methods, known as *ontology matching* techniques, can be found in (Euzenat and Shvaiko 2013).

```
Class: Item

Class: Food
 Subclass of: Item

Individual: mate
 Type: Drink

Individual: coffee
 Type: Drink

ObjectProperty: sells
 Characteristics: Functional
 Domain: Shop
 Range: Item

Class: Drink
 Subclass of: Item

Class: Shop
 SubClassOf: sells min 1

Class: CoffeeShop
 Subclass of: Shop
 SubClassOf: sells only {Drink, Food}

Individual: SandwiChez
 Types: CoffeeShop
 Facts: sells coffee, not sells mate
```

Figure 2.1: A simplified OWL ontology representing coffee shops and the items they sell. It uses the Manchester Syntax. Briefly, mate and coffee are individuals of class Drink, which is a subclass of Item together with Food. A CoffeeShop is a subclass of Shop that sells only food and drinks, and Sandwichez is a coffee shop.

Following the terminilogy in (Bouquet et al. 2004), ontology matching techniques find an *alignment* between concepts in two different ontologies that we will call O_1 and O_2. An alignment is a set of tuples $\langle c_1, c_2, n, r\rangle$ where c_1 and c_2 are concepts in O_1 and O_2 respectively, n is a number that represents the confidence on the mapping, and r is the relation with which the concepts are

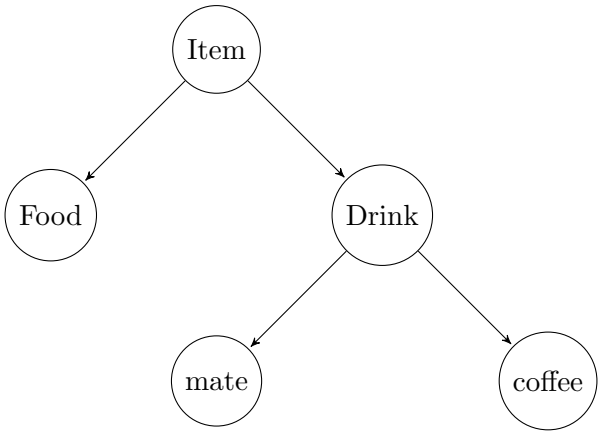


Figure 2.2: Taxonomy representing drinks. Circles are concepts, and the arrows between them represent *is-a* relations.

mapped. The most common relations are equivalence ($\equiv$) and subsumption ($\sqsubseteq$). The former corresponds to two equivalent concepts, while the latter means that a concept is more general than the other one. The alignment is obtained by analyzing and comparing the structure in the two ontologies to be matched. There exist many different ways of doing this, depending on the kind of specification used in the ontologies. According to Euzenat and Shvaiko (2013), techniques usually measure the similarity between individual concepts. Different methods are employed to obtain this measure. Some of them are linguistic, and take into account the syntax, language-specific information, or external resources. Other methods are structural, and look either at the shape of the concept (for example how many attributes it has) or at its relation with other concepts. Then, the tools combine this information to obtain a global alignment between the ontologies. Of course, the success of ontology mapping techniques depends heavily on how much information is available in the ontologies to be mapped and on how similar they are.

Ontology Alignment for Multi-Agent Systems

The problem of semantic interoperability in multiagent systems has been proposed as one of the applications of ontology matching tools (Euzenat and Shvaiko 2013). Indeed, ontology matchers can be used in a simple way to solve the problem of agents that need to interact but speak different languages. Before agents start communicating, an alignment between their ontologies is computed; then agents use it as a translation to understand foreign messages. This simple method, however, has some problems. Mainly, it requires to map complete on-

tologies, whereas it is possible that the interactions agents need to perform use only a small portion of the vocabulary. In this case, mapping all the ontology would be an unnecessary use of resources. In addition, it is possible that agents are not willing to share their knowledge representations. Finally, this idea relies on the quality of the alignment tools that are available to the agents.

Some approaches have tried to solve these issues, proposing more efficient ways of integrating ontology alignments in multiagent communication. For example, to tackle the problem of having to map complete ontologies, van Eijk et al. (2001) propose an on-demand approach in which agents that are performing a task together only map words that are necessary in the interaction. Concretely, instead of aligning all their vocabularies before interacting, agents start sending messages and only look for a mapping when they find a word that they do not know. Another on-demand approach is the one by Besana et al. (2005). This work considers agents in a peer-to-peer network following protocols to perform a task, which is considered to be the *context* of the interaction. The authors explore different ways in which agents can extract information from the task to decide which part of their ontology is worth mapping, by analyzing the probability that it will be necessary later on in the interaction.

Other approaches focus on reducing the amount of information that has to be exchanged to find a mapping. As we mentioned, many ontology matching techniques compute the similarity of two words by comparing the information the ontologies have about them. However, agents may not be willing to share their full knowledge structure for many reasons, including privacy and efficiency. This can be addressed by building techniques that let agents exchange, on demand, just the information that is necessary to map two concepts. These techniques propose interaction protocols that agents can use to find an alignment exchanging only minimal information. An example is the work by (Santos et al. 2016b) and by (Laera et al. 2007), who propose to use an argumentation-based technique to reach a common agreement over alignments is proposed. More concretely, the authors propose a protocol that allows agents to discuss about mappings and choose which alignment they are going to use. An advantage of this method is that the obtained mappings are shared instead of only known to one side, and both agents can take them into account during the interaction.

Finally, some approaches combine the two previous ideas, providing an interaction protocol to find alignments when they are necessary during a dialog. One example is the system proposed by McNeill and Bundy (2007). In this work, misunderstandings during an interaction are considered to be caused by faulty ontologies. The authors propose a technique that agents can use when this happens, to discover how they should repair their knowledge structures to stay aligned.

2.3 Meaning as Relations between Words and Objects

Perhaps the simplest representation of meaning is as a relation between words and things in the world. Under this so-called *extensional view* (Frege 1948), the word *dog* corresponds to the set of dogs in the world. A correspondence between words and meanings is usually known as a *lexicon*. The approaches in this section consider only physical objects in the world as meanings, although this evidently cannot express the complexity of the meaning of some words. The computational models that we describe consider the question of how a community of agents that have a shared environment (that is, that can observe the same objects) can reach an agreement on a common vocabulary to name these objects.

We will discuss two approaches to this problem. The first one considers the emergence of a previously non-existent vocabulary, while the second one discusses how the correspondences between names and objects can be learned. At first sight, these approaches can seem different than the ones we discussed previously, since they deal with the acquisition of a new vocabulary instead of with the alignment between two existent ones. However, it is important to note that these techniques assume that all agents have the same candidate objects. That is, they all agree on which things in the world are to be named. Concretely, learning agents are supposed to have a token for each object, that they align with the unknown vocabulary, which is essentially the same as mapping foreign words with known ones. Put differently, agents in this section are not learning a categorization from scratch, but only naming the parts of an already shared one. The interesting problem of learning categorizations (or concepts) has been discussed, for example, in (Lake et al. 2015, Ontañón and Plaza 2010).

Let us start with a discussion of the emergence of a vocabulary, that is, of how agents that do not share a lexicon can create one from scratch. A well known model of the emergence of a shared lexicon is the one developed over the years by Luc Steels and collaborators (see (Steels 1995) for a first explanation of the technique and (Baronchelli et al. 2006) for a detailed analysis of its dynamics). This work considers agents that share an environment with physical objects, and analyzes how they can, by interacting, create a common lexicon to name them. The process happens in a completely autonomous way, without any centralized control or global synchronization. Agents engage in local interactions, but the lexicon they obtain is global. The approach consists in a game through which agents share their names for a given object. Concretely, each agent has its own lexicon, which is initially empty. In a game, one agent plays as the sender and the other one as the receiver (as in the *signaling games* proposed by Lewis (1969)). The sender points to an object and utters a word it associates with it

(or a random word if the object has no associated name yet). The receiver then analyzes its own interpretation choices for the word. If they include the one that the sender pointed at, the game succeeds and both agents delete any other names for that object; otherwise the receiver updates the possible names with the new word. This simple approach can also model uncertainty, when the perception of the objects that are pointed at is not completely clear. The authors show how, by playing repeatedly, a community of agents can spontaneously agree on a lexicon.

The second group of techniques that we discuss investigates how lexica can be learned. These models are inspired on the question of how children can learn correspondences between words and meanings from observing situations and hearing utterances that describe them (Xu and Tenenbaum 2007). The problem is then how the learner can get to identify which objects in the situation is being described with each word in the utterance. The philosophical problem that is at the background of this research is known as the indeterminacy of translation. Quine illustrates this problem with the famous *gavagai* example (Quine 1960). Suppose you are in a foreign community whose language you do not know, together with a group of natives. You see a rabbit running, and one of the natives points at it saying *gavagai!* The word *gavagai* could mean *rabbit*, but it could also mean *food* or *let's go hunting*. The problem is, then, how an alignment can be decided when many possibilities exist. In a more everyday setting, consider a child observing the environment in Figure 2.3 and listening to the utterance *there is a cup of coffee on the table*. A priori, there is no way to decide if the word *coffee* refers to the computer, the plant, or to the situation in which a thing is on top of another one. It it possible to learn a correspondence between words and things from these undetermined utterances?

The computational models that we review propose a solution to the problem of the undetermined reference: if the same word is observed in different situations, then learners can infer its meaning by reasoning about them. These models, in which many situations are observed before trying to map objects and words, are known as *cross-situational*. A situation consists of a set of objects and an utterance, composed of one or more words. For example, for the situation in the previous picture, the hearer would receive, for example, the objects cup, coffee, computer, plant, table, and notepad, and the utterance *cup, coffee, table*. The objective of the learner is to decipher which word corresponds to which object(s). This can be difficult by observing only the situation above, but if the learner observes also another situation where there is a cup of coffee in a kitchen counter, it may analyze the similarities and differences between both cases.

The way in which learners reason about situations to infer a lexicon depends on the particular work. For example, the approach in Siskind (1996) uses a set of inference rules. Most of the modern work in this direction, however, follows

Figure 2.3: Observed Situation: a table with a cup of coffee, a computer, a plant, and a notepad

a probabilistic approach, in which each lexicon has a probability of being the correct one that is computed based on the evidence provided by the corpus. In general, the models consider Bayesian learners that take into account the corpus as well as priors that represent meaning expectations. This allows to introduce in the model a variety of hypotheses about vocabulary acquisition. Concretely, in its simplest formulation, learners that observe a corpus C of situations compute a probability distribution over their possible lexica L using Bayes rule:

$$p(L \mid C) = p(C \mid L)p(L)$$

The prior $p(L)$ allows to introduce hypotheses about preferences over lexica, or past learning from a previous corpus C'. The probability $p(C \mid L)$ is computed according to hypotheses about how learners relate words with objects. In this way, different theories about language learning can be integrated into the same model, and evaluated on the same datasets. For example, Xu and Tenenbaum (2007) take into account a taxonomic structure relating the concepts that is known by the learners, and assume certain preferences about the level of generality of the concepts denoted by words. Frank et al. (2009) use a model of the speaker's intentions, and Yu and Ballard (2007) incorporate other social cues such as joint attention and prosody.

These approaches are usually evaluated on real corpora of utterances addressed to children, such as the CHILDES database (MacWhinney 2000), measuring how well different hypotheses perform. The area is well developed, and different extensions have been proposed, such as taking into account corrections from teachers to learners (Angluin and Becerra-Bonache 2017) and considering a multi-lingual environment (Zinszer et al. 2017).

2.4 Meaning as Relations between Words and Effects in an Interaction

A third dimension of meaning, which we have discussed in depth in the introduction, consists in the role that words play in an interaction. This idea relates words not to objects (physical or not) in the world, but to actions. More concretely, a word is related to the effects that its utterance provokes. If, for example, when a waiter asks *what do you want to drink?* the customer always answers with the name of a drink, we would say that the meaning of the utterance is the triggering of that response.[1] This take on meaning in rooted on the pragmatist philosophical tradition. In relation to language, pragmatists often propose to focus on behavior, rejecting the *representationalist* view of meaning that relates signs with objects. As put by Dewey, "meaning is established by agreements of different persons in existential activities having reference to existential consequences" (Dewey 1938).

In this sense, we can say someone understands a language if she can interact successfully with other speakers, satisfying with her actions the expectations of her interlocutors about the effects of words. To understand the meaning of a word it is necessary to utter it and see its effect; therefore the approaches that consider this kind of meaning propose to learn an alignment by interacting and observing.

The idea of learning alignments from observing the consequences of utterances was developed in a very general way by Goldman et al. (2007). In this work, agents learn to communicate by maximizing rewards in an environment that can be modeled as a Markov Decision Process (MDP). Very briefly, a Markov Decision Process consists of a set of states, each one associated with some actions that can be performed in that state. When an agent performs an action, there are two consequences: the current state changes, and the agent receives a reward (or punishment). In this way, an agent can move through states performing a sequence of actions and receiving rewards. In the environment proposed by Goldman et al., the actions that can be performed in each state correspond to different interpretations of received messages in a foreign language. By interacting repeatedly and applying simple reinforcement learning techniques, agents learn interpretations that result in better rewards. While their method provides a new, pragmatic view of alignment learning, it is very general, and in many cases expressing the environment as a MDP is not trivial. In an effort to make the approach more concrete, the work in Barrett et al. (2014) studies a version of the problem specially designed for the multiagent, multiarmed bandit problem.

[1] *What do you want to drink?* is, of course, not a word. For now, we will simplify this and consider messages as an indivisible unit. We will come back to this aspect in Chapter 6.

A different approach, that has been proposed in Atencia and Schorlemmer (2012) and discussed in Chapter 1, considers meaning to be related to socially accepted flow of interactions. This is the view that we will adopt in this dissertation. Intuitively, interaction contexts dictate what it is possible to say at each moment. For example, talking about bread in a bakery makes more sense than talking about screwdrivers. These notions are considered to be culturally shared; that is, all agents, even when they use different vocabularies, have the same idea about how interactions should be performed. A correct alignment between different vocabularies is one that allows agents to interact without violating the rules of the interaction.

Concretely, in Atencia and Schorlemmer (2012) the authors present a method called Interaction-Situated Semantic Alignment (I-SSA), that explores how the context of the interaction can be exploited to obtain an alignment. They consider agents communicating with interaction protocols, and develop a method that allow them to learn mappings from the experience of interacting.

Since I-SSA is the technique on which our work is based, let us explain it in more detail. The original formulation of the method considers two agents that interact to perform a task together. Each agent has its own protocol that determines how the task can unfold. Concretely, protocols are finite state machines that define the flow of an interaction by determining, at each state, which messages can be sent. For example, the protocol in Figure 2.4 specifies an interaction to buy drinks between a customer (C) and a waiter (W). The authors assume that both agents have similar protocols, but with labels in different languages. For example, the customer could have the protocol we presented, while the waiter could follow one in Italian. When interacting, each agent utters words in its own vocabulary, so its interlocutor needs to find a way of interpreting the messages it receives. This is achieved by letting each agent have a value for each mapping between a local and a foreign word, that represents the confidence on that match. When an agent receives a message, it chooses an interpretation according to those values.

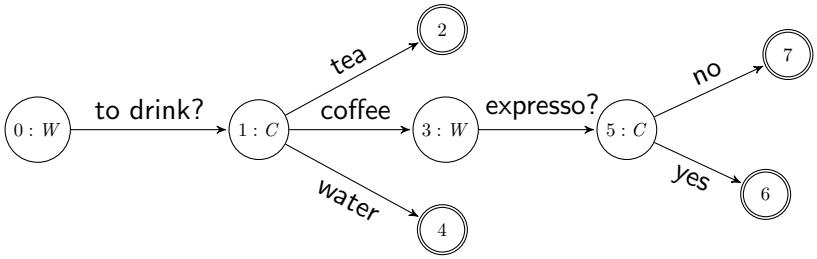


Figure 2.4: Ordering drinks protocol. Each state is labeled with its corresponding speaker, and each label with the message that can be sent.

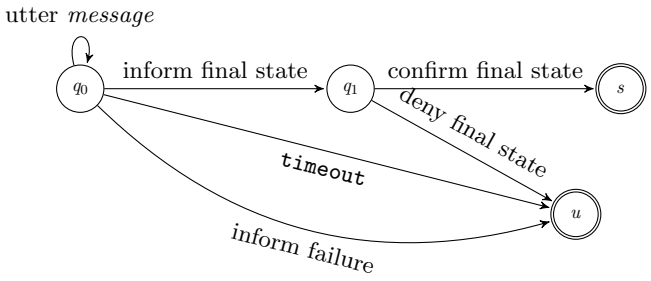


Figure 2.5: I-SSA alignment protocol. Agents are in state q_0 while neither of them has arrived to a final state in their protocols. When one does, they exchange information to determine if they both finished. The state s is a successful interaction, and u is an unsuccessful one. These can also be achieved by informing a failure (usually produced when an agent cannot interpret a message) or when there is a timeout.

The values for mappings are updated from the evidence observed when interacting. Concretely, I-SSA considers an interaction to be successful when both agents reach a final state at the same time, and all mappings in a successful interaction are correct. To identify when this happens, agents follow a meta-interaction, specified in Figure 2.5, to decide if they both reached the end. If they did, they increase the value of the mappings they made. If the interaction fails without this happening, they end the interaction without any updates. Atencia and Schorlemmer show how, using this technique, agents can converge to an alignment that allows them to finish interactions successfully most of the times. They perform an analysis of which factors make the convergence faster or slower, comparing protocols with different shapes and sizes, and varying vocabulary sizes.

2.5 Discussion

The techniques we presented in this section tackle the problem of how interacting agents can understand each other even if they originally do not share the same language, using information corresponding to three different dimensions of meaning. All the approaches are, in a way, contextualized by the resources that are available. In the first dimension of meaning, which relates words between each other, the ontologies determine the kind of alignment that is obtained. If, for example, the ontologies describe relations holding in the domain of cafeterias, then some of the obtained alignments may not be useful for other situations. In the second dimension, which relates words with things in the world, the learned

correspondences are determined by the type of situations that are observed. In the third dimension, which relates words to their use in interaction, the context is given by the specification (in protocols) of the tasks that are performed.

We will now discuss three aspects about the techniques that we reviewed, that we preferred to address jointly rather than for each dimension by separate. First, we focus on the elements that are assumed to be shared in the approaches we described. Second, we discuss possible combinations between the different dimensions. Third, we put the contributions of this dissertation in context, relating them to the previously discussed techniques.

2.5.1 What Needs to be Shared

All the vocabulary alignment methods we discussed assume different levels of shared elements between the agents that need to align their vocabularies. We now discuss four different aspects in which agents need to agree.

Conceptualization of meaning. Although it may seem evident, it is important to point out that it is always assumed that agents know what dimension of meaning they are working with. That is, that words are related in ontologies in the first case, mapped with objects in the second, and related to the flow of the interaction in the third one. If the learners in the second dimension do not know that words map with objects, the learning techniques that we discussed would be pointless.

Relation between meaning and the use of words. For example, in the second dimension all agents agree that the words in the utterance that is related with a situation refer to objects in the scene, and not to objects that are *not* there. In the case of I-SSA, agents assume the uttered words respect the protocol.

Concrete elements to ground the meaning of words. Agents need to share something concrete they can use to decide whether two words are equivalent or not. In the first dimension, agents sometimes need a common metalanguage to discuss the meaning of words, or a set of shared ones to describe others with them. In the second dimension they need a shared environment, with objects they can both observe. In I-SSA they need to share the observation of when an interaction ends, while for the approach that models learning on a MDP they need a shared reward.

The actual meaning of (some) words. The previous three were only the formal requirements of each method; the ones that are necessary to infer meaningful alignments are more restrictive. Concretely, agents need to have some-

thing in common in their meaning structures to obtain useful mappings. In the first dimension, the ontologies need to be similar; if the structures are radically different (for example, if they agents are trying to map an ontology about medical concepts with one for cooking terms) they will hardly obtain a good alignment. For the second case this corresponds to the shared categorization of objects to be named. In the case of I-SSA, they need to share the protocol structure, or the same idea of how tasks are performed.

2.5.2 Combining Sources of Meaning

As we have already discussed, the three approaches we reviewed are only partial representations of the semantic knowledge that may be available. Although we saw that information about an alignment can be extracted from the different dimensions separately, a richer background could allow for a richer inference process. Even without introducing any new meaning representations, an immediate question is how the three approaches that we proposed can be combined, obtaining an alignment method that extracts information from all of them. This problem, however, is not trivial. This is because the different dimensions of meaning are not independent, but it is difficult to specify how they are related. Therefore it may not make sense to simply combine a numerical similarity degree obtained from each technique. Instead, a meaningful combination would use the relation between the three dimensions of meaning. For example, to combine the dimension of relations between words and the one of relations between words and objects, it would be necessary to know how ontologies and pragmatic aspects are related, for instance knowing what kind of concepts are more likely to be used in a given task. In our work we will only consider a simple approach that integrates any kind of external alignment that is obtained with other methods (see Chapters 3 and 4).

There exists work that combines meaning elements in these different categories. For example, the work by van Diggelen et al. (2006) uses both the first and second dimensions of meaning. The authors present a technique to build a common ontology that creates definitions on demand for words that are necessary during an interaction, in two layers. First, the agents try to explain the concepts with other shared words; if this fails they resort to physical information. A subtler combination is proposed by Euzenat (2014) (extended in (Euzenat 2017)), who considers the problem of repairing ontologies by using cultural information. Concretely, he proposes a way of merging the information obtained from interacting with the ontological information that each agent has. This is achieved by defining a method that allows agents to see physical examples for each word, and proposing different ways of updating their local ontologies to make them more similar. Another combination of this kind are the models of cross-situational word learning that consider priors related with

a taxonomical structure (Xu and Tenenbaum 2007). Even considering the same type of meaning representation, the integration of similarity measures is not straightforward. For example, Li et al. (2009) discuss how to merge different ontology matching techniques, showing that if it is not done carefully, the combination can be worse than applying each method separately. They propose a system that automatically determines which techniques should be used in each case, depending on the characteristics of the ontologies that are given as input. This solution, however, uses decision making rules that are developed ad-hoc for each of the existing techniques, which harnesses its generality.

A different way in which techniques can be combined is in their evaluation. Normally, alignment techniques are evaluated with respect to a human-crafted *reference* alignment that is considered to be correct. However, this alignment may be unavailable or be difficult to obtain. If different semantic sources are available, the alignments obtained from them can be compared with each other. More interestingly, an alignment obtained from one semantic resource can be tested using a different approach to see whether it is consistent. For example, an ontology alignment can be tested on the second dimension of meaning by observing if it is useful to name objects that actually appear in a given situation. Later in this dissertation we will present a way of evaluating ontology alignments using an interaction.

2.5.3 Our Contribution in Context

As we already anticipated, in this dissertation we will focus on the third dimension of meaning; more concretely, on the idea of aligning vocabularies by observing how they are used in interactions. This dimension is by far the least developed of the ones we presented, despite being, as we discussed in the introduction, a useful way of representing how words are used in dialog. Importantly, using this information to align vocabularies means not resorting to anything outside of the communication situation. While the other two dimensions assume the existence of structured semantic knowledge and a physical world respectively, in our case agents are only assumed to have a specification of how the interaction is performed, which is already necessary to interact meaningfully.

In the following chapters we will develop different aspects of the I-SSA technique. We incorporate elements from the other approaches discusses in this chapter to the basic idea proposed in Atencia and Schorlemmer (2012), to make it more efficient or more flexible. In Chapter 3 we discuss the combination of external ontology alignments obtained with methods that adopt the first dimension of meaning with an interaction-based technique. To do so, we reformulate the idea in I-SSA as a reinforcement learning problem, in the style of the one proposed in (Goldman et al. 2007). In Chapter 4 we apply the idea of learning from the interaction context to much less restrictive protocol specifications.

This makes it necessary to adopt a cross-situational approach like the ones we described when discussing the second dimension of meaning, since alignments are not fully determined with one task anymore. The updating techniques we use in this case are also similar to the ones we explained. Using these well-known learning paradigms, we are able to explore various possibilities and to decide which ones are best suited for different situations.

Chapter 3

Integrating Vocabulary Alignments and Interaction-Based Techniques

In Chapter 2 we described different approaches to vocabulary alignment, each corresponding to one of three representations of meaning. In particular, we described the *interaction-based vocabulary alignment* idea, which consists in inferring an alignment from repeated interaction by leveraging information from the specification of tasks. In this chapter we present an improvement of the existing interaction-based vocabulary alignment techniques and we analyze how this approach can be combined with external resources. We consider the two aspects of combination that we described in the previous chapter. First, we analyze how external alignments can be integrated with an interaction-based technique to obtain a method that takes advantage of extra information, and show that is also robust against alignments with erroneous information. Second, we propose ways to use interaction-based techniques as evaluation frameworks for existent alignments.

3.1 Introduction

One of the most appealing aspects of the interaction-based approach to vocabulary alignment is that it does not use any external resources except for the task specifications that agents already need to interact. The approach considers meaning to be only determined by how a word is used in interactions, without taking into account any semantic or syntactic resource. This position is taken to the extreme; for example, suppose an agent receives a message m_1 in a foreign language, and has to choose a local word to interpret it. Even if it is expecting a local message m_2 that looks exactly equivalent to m_1, the agent will not favor

that interpretation just for this reason. Although this may seem to be a radical choice, it makes sense in the context of interaction-based alignment: m_1 and m_2 are not necessarily similar from the point of view of the interaction semantics, even if they are written in the same way.

A skeptical take on external sources of meaning can be useful because it avoids possibly incorrect mappings. For example, it can avoid confusion caused by the pairs of words commonly known as *false friends*, such as *actually* and *actualmente* (currently) in English and Spanish or *salir* (to get out) and *salire* (to go up) in Spanish and Italian. The other –more practical– advantage of not using external knowledge is that the resulting method does not assume the existence of other resources, resulting on a more flexible and cheaper technique that can be applied for any kind of interaction domain and language combination.

The downside of considering only the information in the interaction specifications is that the resulting method can be very slow when the protocols are large. I-SSA, the method introduced by Atencia and Schorlemmer (2012) that we described in Chapter 2, relies on interpreting foreign words randomly until it finds a sequence of correct mappings that leads to a successful interaction. This can be very slow if protocols are large, since there are many decisions to make, each one with many possibilities. To make things worse, a mistake in one of them makes the whole interaction fail. In this situation it would be helpful to use extra information to avoid making choices randomly. Although false friends exist, the possibility of finding erroneous mappings may not be reason enough to dismiss all the available extra information. Intuitively, an English speaker in a bar in Italy would rarely overlook the fact that *caffé* sounds and looks similar to *coffee* – in fact, she should not overlook it.

In this chapter we focus on the question of how external alignments can be taken into account in an interaction-based vocabulary alignment. We propose a skeptical integration, in which external alignments are assumed to contain some wrong mappings and we try to minimize their effect. Alignments that are obtained with automatic techniques, can never be completely trusted. First, the methods themselves always have a precision lower than 100% for technical reasons (see the work by Cheatham et al. (2015) for a recent evaluation of ontology matching techniques, and the papers they refer to for evaluations of vector-based techniques). In addition, which mappings are considered correct can change from one application context to another, so an alignment with high precision for a context can be not useful for interactions in a different environment. In line with the general spirit of interaction-based alignment techniques, we still consider that a correct alignment is one that lets agents interact, and we study how agents can find it more rapidly with the help of external tools.

We propose two approaches to integrate external alignments. The first one considers these alignments as previous knowledge, and learns on top of them. The second approach is more complex, and it uses the alignment information in

a more intelligent way. To define these two techniques, we first reformulate the I-SSA technique, presenting it as a reinforcement learning problem in which it is simple to introduce previous knowledge. Reinforcement learning (Sutton and Barto 1998) studies how agents can learn to make decisions in a particular kind of space. Briefly, the goal is to learn how to maximize a reward, by interacting with the environment and observing the consequences of each action. Formulating the interaction-based vocabulary alignment techniques in these terms also allows to better understand the strengths and weaknesses of the I-SSA technique, and provides possible solutions for the latter. Concretely, it becomes evident that the main reason why the learning is slow corresponds to the *delayed reward* problem in reinforcement learning. That is, agents only know if they chose good or bad interpretations at the end of the interaction, when they find out whether the interaction is successful or not. If it is not (which happens most of the times) agents do not know at which point they made a mistake, which makes it difficult to learn anything.

As we discussed in Chapter 2, another way of combining two different alignments is to use one to evaluate the quality of the other. Typically, this is achieved by considering one of the alignments as a gold standard, and by measuring how similar the other alignment is. In this chapter we propose to use an interaction situation to evaluate external alignments. Concretely, the idea is to determine if an alignment would be useful for agents that need to perform a particular interaction task, but speak different languages. We define the quality measures that are necessary to perform this evaluation, gaining insight on which type of alignments are best for interacting agents.

Summarizing, the objective of this chapter is threefold:

1. It defines the communicative situation that we consider in a large part of the dissertation and presents useful definitions and methodologies. It also introduces central examples that will serve as illustration.

2. It provides a novel formulation of interaction-based alignment techniques as a learning problem, allowing to use the tools that exist for this kind of problems.

3. It explores the possible combinations between interaction-based alignment techniques and other methods to align vocabularies.

In Section 3.2 we describe the general communicative situation that we consider and introduce examples that we will later use. Section 3.3 presents our version of the I-SSA technique, formulated as a reinforcement learning method. In Section 3.4 we discuss how external alignments can be used to interact successfully. In Section 3.5 we discuss two different ways of combining both sources of information, that we evaluate in Section 3.6. Section 3.7 presents our approach to alignment evaluation, and we finish with a discussion in Section 3.8.

3.2 Performing Tasks by Interacting

Throughout most of this thesis (Chapters 4, 6 and this one) we will work with the following scenario. Consider agents a_1 and a_2 who need to collaborate to perform tasks together. For example, a task could be *ordering a drink at the pub*. They both agree on some aspects of the tasks they can perform, for example:

- They both know the set of tasks that can be performed together.
- Whenever they are performing a task, they agree on which task it is (that is, it never happens that a_1 is trying to buy a drink from a_2, but a_2 thinks they are making pancakes together).
- They both know which role each of them is playing while performing the task (for example, they agree on who is the customer an who is the waiter).

Agents perform tasks together by interacting, and they interact by sending messages to each other. This idea is not new; the notion of performing actions by speaking is at the core of the Speech Act Theory developed by Austin (1975), Searle (1969), and Lewis (1969). These authors suggest that language is not only used to describe the world, but also to change the environment. For example, when someone says *"I will have a coffee"* in a bar she is doing something more than stating facts. In our case we take this idea to an extreme; the only way of performing tasks together that our agents have is by sending messages. Of course, in some cases there may exist other aspects in an interaction apart from strictly verbal ones. If agents are involved in a buying and selling interaction in a real pub, they could use mimics to explain themselves, for example by pointing at the drink they want. Additional information sources have their own requirements in terms of available communication channels and agent capabilities. For example, to be able to mimic, agents need to have access to a common physical environment or a video camera, to see what is happening. Regarding the agent's abilities, the sender needs to be able to point, and some visual capability is required for the receiver. We consider a general communication framework in which messages could be construed in different ways, without focusing on the implementation details. This a level of abstraction that allows us to fit many different communicative situations. Sometimes, for simplicity, we will talk of agents *speaking*, but still encompassing other possible ways of communication.

For each task, each agent has a local (that is, not necessarily shared) specification that describes how it should be performed. Since tasks are performed by interacting, this specification contains details about the messages that should or should not be sent to obtain the goal. Concretely, a specification is composed of a set of states where each state is associated to a set of messages that can be said, and that move the interaction to another state. Of course, there exist

multiple ways in which this information could be specified; we discuss different ones in the following chapters. Central to our investigation is that we assume that, at least to some extent, agents have the same idea about how a task should be executed. We will come back to this later and clarify this notion for each of the proposed specifications.

Since all the communication is based on message passing, it is crucial that agents understand each other's messages. We consider a case in which this condition does not necessarily hold, since agents may speak different languages. Although languages can be very sophisticated systems, in this dissertation we will focus only on *vocabularies*. A vocabulary is the finite set of words an agent is allowed to use in its messages, and we will write V_1 and V_2 for the vocabularies of agents a_1 and a_2 respectively. Each agent can organize its vocabulary in its own way, with additional structure that makes it a taxonomy of words, or even a full fledged ontology specifying the intended meaning of the words of the vocabulary. We do not need to commit to any of these assumptions in our work.

The problem we approach, then, can be summarized as follows. Agents a_1 and a_2 need to perform a set of tasks together. They both agree on how to perform the tasks, but they speak different vocabularies V_1 and V_2 respectively. How can they reach a level of understanding that allows them to interact meaningfully? Note that, for simplicity, we are restricting the problem to only two agents interacting with two languages. As discussed in (Atencia 2010), an interaction between more than two agents can be transformed into various two-agent interactions, as long as there are no broadcast messages. In the simple version of our technique, agents do not use any reasoning on their interlocutors, and the techniques are performed individually. Therefore, it is not important if they speak with different agents, as long as they all use the same foreign language. If there is more than one foreign language, agents need to know which one their interlocutor is using, and the technique must be performed in parallel for each one.

Ultimately, the problem of interacting with someone who speaks a different language lies on how to interpret messages. If whenever an agent receives a message in a foreign vocabulary it chooses as an interpretation a local message that is equivalent for that particular situation, then it can be said that the agent understands the foreign language (see Chapter 2 for details on this conceptualization of language understanding). Formally, we consider the situation in which agent a_1 receives a message with a word $v_2 \in V_2$ from agent a_2. In this particular situation a_1 might be expecting any message in a set of *possible messages*, which is determined by its specification. Our objective is to provide a_1 with a way of deciding which word between the possible ones it should choose as an interpretation, to be able to continue the task. We use a probabilistic technique: each agent has a confidence distribution that expresses their beliefs about how words should be interpreted. Along this dissertation we will explore different

ways of updating this confidence distribution.

In this chapter we discuss four different techniques to update the confidence distribution on mappings. First, we propose a simple interaction-based vocabulary alignment technique based on the ones proposed in (Atencia and Schorlemmer 2012). Second, we show how external alignments can be used to build a translation. Finally, we show two ways of combining these approaches. In all cases we explain the methods in two steps:

1. Agents compute a distribution ω that represents their confidences in mappings between the foreign and local vocabularies

2. Agents use the distribution ω to choose how to interpret foreign messages

Now, let us introduce two task examples that we will use throughout the dissertation.

Example 1: Travel Agency In the travel agency example, a customer interacts with a travel agent to book some tourist service. They both speak English, but they nevertheless disagree on the vocabulary, because the travel agent speaks American English while the customer uses its British variant. For example, the travel agent says OneWay while the customer says Single to refer to a ticket without a returning journey. In the interaction, the Customer first makes some choices (for example if it wants to book a journey or an accommodation) and inputs data (such as dates and city to visit). Then the travel agent answers with a set of offers, or by stating that there are no available options. This example was adapted from the one in (Atencia 2010) and will be useful in the first part of this chapter, since we have an alignment between words that was obtained with an ontology matching tool.

Example 2: Ordering drinks In this example a customer interacts with a waiter to order drinks. The customer can ask for beer or wine, and the waiter then asks more specific questions according to the chosen beverage. However, in this case, the customer speaks English and the waiter Italian. This simpler example will be used in the last part of this chapter and in the following one.

3.3 Learning from the Experience of Interaction

In this section we revisit the work by Atencia and Schorlemmer (2012), which considers agents following interaction specifications formalized with finite state machines and provides a technique to learn an alignment between their vocabularies from the experience of interacting repeatedly. We extend the mentioned work in two ways. First, we introduce more expressive specifications that can

model more complex interactions. Concretely, we propose a new termination condition. Second, we reformulate this alignment approach to express it as a reinforcement learning problem. This will be useful to introduce, in Section 3.5, the novel methods we propose to integrate external knowledge into the technique.

In what follows, we first define the specifications and some useful relations between them. Then we describe the dynamics of the interactions of agents following them. Finally, we introduce a learning method that infers an alignment from repeated interaction.

3.3.1 Interaction Models

As in the work by Atencia and Schorlemmer (2012), we consider interactions that are specified with finite state machines, which we will call *interaction models*. Concretely, an interaction model is a description of all possible interactions that can be performed. It is composed of a set of states and a transition function which determines, for each state, what messages can be sent and to which state they lead. An interaction finishes when it reaches a final state. Each final state is labeled with a set of properties that represent different interaction outcomes. For example, a final outcome in an interaction to order drinks could be *the waiter gives a coffee to the customer*. This element, not present in (Atencia and Schorlemmer 2012), will later play a role in the alignment inference part. If there are no jointly observable outcomes this set will be empty, which corresponds to the original formulation. In what follows, a vocabulary is a finite set of words and state properties are propositional variables.

Definition 3.1. Given two agents a_1 and a_2, a vocabulary V and a set of state properties SP, an *interaction model IM* is a tuple $\langle Q, q^1, \delta, F, \rho, speaks \rangle$ where:

- Q is a finite set of states,
- $q^1 \in Q$ is the initial state,
- $F \subseteq Q$ is the set of final states,
- $\rho : F \to \mathcal{P}(SP)$ is a function assigning a subset of state properties to each final state,
- $speaks : Q \setminus F \to \{a_1, a_2\}$ is a function assigning to each non-final state its sender agent, and
- $\delta : Q \setminus F \times V \nrightarrow Q$ is a partial function called the *transition function.* ■

Note, first, that, unlike traditional FSM definitions, we do not allow agents to continue speaking in final states. Second, that we do not specify any particular

turn-taking pattern. We do require that, for each state, all messages labeling transitions from this state share the same sender agent, who is determined by the *speaks* function. A sequence of messages sent by agents is an interaction.

Definition 3.2. Given an interaction model $IM = \langle Q, q^1, \delta, F, \rho, speaks \rangle$ over a vocabulary V, an *interaction* I is a string in V that is accepted by IM. That is, a sequence $v^1, \ldots, v^n$ of words in V such that there exists a sequence $q^1, \ldots, q^{n+1}$ such that q^1 is the initial state, $q^{n+1} \in F$, and for all $1 \leq i \leq n$, $\delta(q^i, v^i) = q^{i+1}$. We will call such a sequence of states a *state interaction*. More concretely, a state interaction I^Q is a sequence $q^1, \ldots, q^{n+1}$ such that $q^n \in F$, and for all $1 \leq i \leq n$, there exists $v \in V$ such that $\delta(q^i, v) = q^{i+1}$. ■

Typical operations over sequences can be applied to interactions. In this chapter we will use the indexing operation: $I(i)$ is the element in the i-th position of I. Note that the definition of the transition function δ forces our interaction models to be deterministic. That is, on each state there is only one transition labeled with each word. Non-deterministic finite state machines can always be transformed into deterministic ones (Hopcroft and Ullman 1979). Considering deterministic automata means that each interaction in an interaction model can be associated with only one state interaction representing the states that are visited to complete it. A state interaction can be associated with many different interactions, if there are many transitions between two states q and q'. We will call $Q(I)$ to the state interaction associated with an interaction I, and $V(I^Q)$ all the interactions associated to the state interaction I^Q.

We assume the language to specify state properties SP is shared; that is, agents can observe the same interaction outcomes. However, they do not necessarily use the same vocabulary to send messages to each other. Concretely, we consider agents a_1 and a_2 that use interaction models IM_1 and IM_2 over possibly different vocabularies V_1 and V_2 respectively. The set of state properties SP is shared, and both interaction models are defined over the same agents.

Examples of interaction models are the ones in Figure 3.1, that specify the interactions for the travel agency scenario described in the previous section. The figure shows the specifications for the travel agent (TA, with interaction model IM_{TA}) and the customer (C, with interaction model IM_C). Each state is labeled with the ID of the speaker, and each transition with the message it sends.

Since each agent uses its own interaction model and sends messages in its local vocabulary, they need a way to decide how to interpret the foreign words that they receive. The goal of the techniques we present here is to provide agents with a translation between their vocabularies that they can use to finish their tasks successfully. We first need to define what it means to interact successfully and what kind of alignments can be used for this purpose. The notion of *communication product*, which describes the combination between two interaction

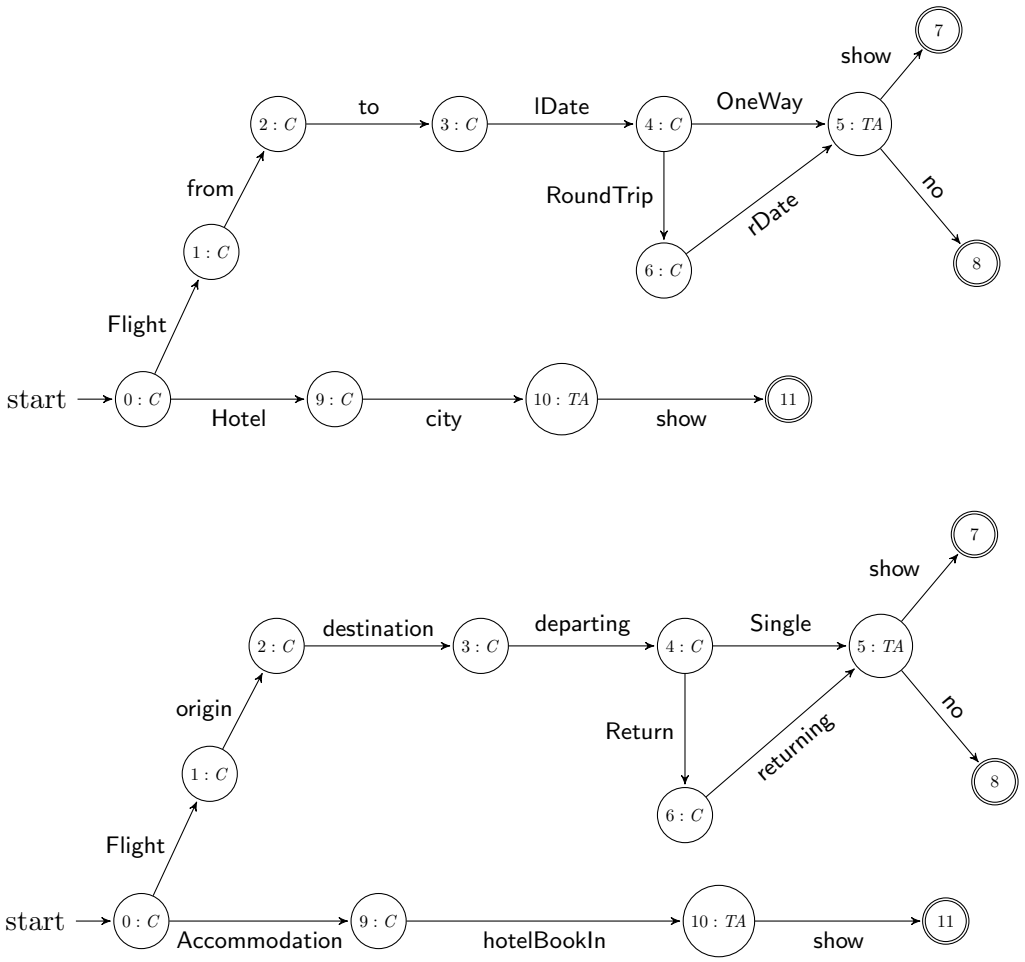


Figure 3.1: Interaction Models IM_{TA} for the Travel Agent (above) IM_C for the Customer (below)

models, will be useful for these purposes. Concretely, the communication product between two interaction models describes all possible interactions between agents who use them, considering the possible interpretation of the messages. From now on, let $IM_1 = \langle Q_1, q_1^1, \delta_1, F_1, \rho_1, speaks_1 \rangle$ and $IM_2 = \langle Q_2, q_2^1, \delta_2, F_2, \rho_2, speaks_2 \rangle$ be two interaction models over the same agents $\{a_1, a_2\}$, the same state properties SP, and possibly different vocabularies V_1 and V_2.

Definition 3.3. Suppose that the interaction models IM_1 and IM_2 are such that $speaks_1(q_1^1) = speaks_2(q_2^1)$. Extending the definition in (Atencia and Schorlemmer 2008), the *communication product* of IM_1 and IM_2 (noted $IM_1 \otimes IM_2$) is an interaction model $\langle Q, q^1, F, \delta, \rho, speaks \rangle$ over a language $V = V_1 \times V_2$, a set of agents $\{a_1, a_2\}$, and $SP = \{success, failure\}$, and such that:

- $Q = Q_1 \times Q_2$
- $q^1 = \langle q_1^1, q_2^1 \rangle$
- $\langle q_1, q_2 \rangle \in F$ iff either $q_1 \in F_1$ or $q_2 \in F_2$, or both are not final and $speaks_1(q_1) \neq speaks_2(q_2)$, that is, if either one of the states is final, or they have different senders
- $\delta : Q \setminus F \times (V_1 \times V_2) \to Q$ is defined as follows: $\delta(\langle q_1, q_2 \rangle, \langle v_1, v_2 \rangle) = \langle q_1', q_2' \rangle$ iff $\langle q_i, v_i \rangle \in dom(\delta_i)$ and $\delta_i(q_i, v_i) = q_i'$ for $i \in \{1, 2\}$
- $speaks : Q \setminus F \to \{a_1, a_2\}$ is defined as follows: $speaks(\langle q_1, q_2 \rangle) = speaks_1(q_1) = speaks_2(q_2)$
- For $\langle q_1, q_2 \rangle \in F$, $\rho(\langle q_1, q_2 \rangle) = \{success\}$ if $q_1 \in F_1$ and $q_2 \in F_2$, and $\rho_1(q_1) = \rho_2(q_2)$. It is $\{failure\}$ otherwise. ■

The communication product represents, in one joint finite state machine, how agents move in their local interaction models. If agents are in a state $\langle q_1, q_2 \rangle$ in the communication product, then a_1's local state is q_1 and a_2's local state is q_2. The labels of transitions represent a sent message and how it was interpreted. That is, a transition $\langle v_1, v_2 \rangle$ from a state where a_1 speaks means that a_1 is sent and a_2 interprets it as v_2. If a_2 is the sender, v_2 is the sent message and v_1 the interpretation. With this construction, we can easily obtain all possible interactions between agents with two interaction models.

Definition 3.4. A state interaction in the communication product $IM_1 \otimes IM_2$ between IM_1 and IM_2 is *successful* if it ends in a state q such that $\rho(q) = \{success\}$, and it is *unsuccessful* if $\rho(q) = \{failure\}$. An interaction I in $IM_1 \otimes IM_2$ is successful if and only if $Q(I)$ is successful. ■

We mentioned that we were going to consider agents that had the same notion of how to perform a task. Now we have all the elements to define this notion formally. Intuitively, agents share the knowledge of how to perform the task if there is a way to map the states in their interaction models in a way that translated interactions are always successful.

Definition 3.5. The interaction models IM_1 and IM_2 are *compatible* if there exists a bijective mapping $\gamma : Q_1 \to Q_2$ between their states such that, for each state interaction in IM_1, all the interactions associated with $\langle q, \gamma(q)\rangle$ are successful interactions in the communication protocol, and for each state interaction in IM_2, all interactions associated with $\langle \gamma^{-1}(q), q\rangle$ are successful interactions in the communication protocol as well. ■

If IM_1 and IM_2 are compatible, then there exists a translation that agents can use to interpret the messages that they receive and always finish the interaction successfully. This translation, however, does not map words between each other. This is because a word v_1 could be mapped to different words in different states. Intuitively, words do not need to mean the same in different points of the interaction. For this reason, it is necessary to know how to interpret each word on each particular state.

Consider all successful interactions in the communication product $IM_1 \otimes IM_2$. These are sequences in $(V_1 \times V_2)^*$. Since the original protocols are deterministic, the communication product is as well, and therefore each successful interaction I is associated with a unique state sequence $Q(I)$. Now consider the relation $T : Q \times V_1 \times V_2$ such that $\langle q, v_1, v_2\rangle \in T$ if and only if there is an index i and a successful interaction I such that $I(i) = \langle v_1, v_2\rangle$ and $(Q(I))(i) = q$. This implies that following the transition labeled with $\langle v_1, v_2\rangle$ in state q, a successful interaction can be obtained. According to Definition 3.5, if IM_1 and IM_2 are compatible, there exists a bijective function that maps states in successful interactions. Therefore each state q in the relation T corresponds uniquely to a state $q_1 \in Q_1$ and a state $q_2 \in Q_2$. Thus, the translation T can be written as $T_1 : Q_1 \times V_1 \times V_2$ and $T_2 : Q_2 \times V_1 \times V_2$. These are the translations agents need to follow to always perform successful interactions. In the following sections we show how they can find these translations.

3.3.2 Interaction Dynamics

Consider again agents a_1 and a_2 following IM_1 and IM_2 respectively. Suppose that IM_1 and IM_2 are compatible, in particular under a pragmatic translation T. While interacting, each agent maintains a local state that corresponds to where it is in its own interaction model. The global state, that corresponds to the one in the communication product, is not observed by them. Agents decide what to say and how to interpret received messages based on a notion of *possible*

words. Intuitively, a word is possible in a state if it labels one of its outgoing arrows.

Definition 3.6. Consider an interaction protocol $IM = \langle Q, q^1, \delta, F, \rho, speaks \rangle$ over vocabulary V. A *possible word* in a state $q \in Q$, is a word $v \in V$ such that $\langle q, v \rangle \in dom(\delta)$. We will call $U(q)$ the set of possible words in q. ■

We will take the perspective of agent a_1 to explain the techniques, but everything is analogous for a_2. Agent a_1 starts the interaction in the initial state q_1^1. Whenever the interaction is in a state $q \in Q_1$ for which $speaks_1(q) = a_1$, the agent chooses a word from $U_1(q)$ to send to its interlocutor. We do not specify how this utterance is chosen; in the implementation for our experimental evaluation (see Section 3.6) it is a random word. If instead $speaks_1(q) = a_2$, a_1 will wait to receive a message from a_2. Since the received word is from V_2, a_1 will need to interpret it in the context of that particular interaction state, according to IM_1. That is, it will choose an interpretation from $U_1(q)$, where U_1 returns the set of possible messages for IM_1. Whenever an agent decides on an interpretation, it remembers it, keeping a record of the mapped pairs as a sequence $(q^1, {v_1}^1, {v_2}^1), \ldots, (q^n, {v_1}^n, {v_2}^n)$. Tuple (q, v_1, v_2) means that the agent interpreted v_2 as v_1 in q (or v_1 as v_2 if it is a_2). Note that this sequence does not include all the states visited by the agent in the interaction, but only those where it was the receiver.

By uttering and receiving messages, agents move along their interaction protocols. As in the original I-SSA technique, we assume agents have a way of communicating properties about the states they are in. Concretely, each time a message is sent and received the agents communicate whether their state is final and, if it is, its state properties. This communication is an abstraction; they could also be able to obtain that information by observation (for example, if they know the interlocutors leave when they finish the interaction). As expressed in the communication product, an interaction is considered to be successful if both agents communicate being in a final state with the same state properties at the same time. The communication fails (and ends) in any of the following cases:

- One agent has reached a final state and the other one has not.
- They both reached final states, but with different state properties.
- They disagree on who's speaking turn it is, either because:
  - they both try to speak at the same time, or
  - none of them speaks for a certain period of time.

3.3.3 Alignment Technique

To interact successfully with each other, agents need to discover the pragmatic translation under which their interaction models are compatible. In this section we describe a method, adapted from the one in (Atencia and Schorlemmer 2012), that lets agents learn the pragmatic translation from their repeated interaction. Briefly, agents observe which interpretations of foreign words make their interactions end successfully. These mappings by definition belong to the pragmatic translation, so they learn to always prefer them.

In (Atencia and Schorlemmer 2012), this technique was explained in a simple way. Agents kept a value for each mapping between foreign and local words; the value for a particular mapping was increased when it was chosen in a successful interaction. This ad-hoc approach is enough to explain how the technique works, however, we will see that other formalizations can be useful to develop extensions. Concretely, we define our situation as a reinforcement learning problem, using their standard concepts and notation. Reinforcement learning (Sutton and Barto 1998) is a way of approaching learning problems in a particular environment called a Markov Decision Process. The learning model consists of a set of states. Each state is associated with a set of possible actions that change the state, and to a reward. The objective of the agents is to learn which actions are best in each state to obtain the highest reward, and this is done by interacting (in a smart way) with the environment, observing the consequences.

In our case, the objective is to learn the pragmatic translation. Let us first express the environment as a learning model. Since a_1 needs to learn which interpretation is good for a word it has received in a particular state, the states of the learning model will be pairs $\langle q, v_2 \rangle$, where $q \in Q_1$, $\mathit{speaks}(q) = a_2$, and $v_2 \in V_2$. In that situation, a_1 can choose how to interpret v_2 from the set of possible messages, therefore the set of actions for a state $\langle q, v_2 \rangle$ in the learning model are the words in $U_1(q)$. Let us make two remarks. First, we are not including states in which a_1 speaks in the learning model, because it does not need to learn any interpretation in those. The pragmatic translation will be independent of the messages it utters. Second, agents do not know the learning model a priori, since they do not know which messages their interlocutor can utter. We will use methods that do not require agents to know the model.

Our objective is to estimate the action values of choosing each word as a mapping in a particular state, that is the confidence in that v_2 should be interpreted as v_1 in q. These values are represented by a confidence distribution that we will call a *situated alignment*.

Definition 3.7. A *situated alignment* for agent a_1 who interacts with a_2 is a partial function $\omega : Q_1 \times V_1 \times V_2 \nrightarrow [0, 1]$. ■

The situated alignment is updated with the interaction, according to what

the agent observes. In this section we present a simple updating technique. All the interpretation decisions made in successful interactions are updated adding a value of 1, and those in unsuccessful interactions are not updated.

Experience Approach (exp).

We are first going to explain how the situated alignment is obtained, and then how it can be used to decide how to interpret words. We will call $\omega_{\texttt{exp}}$ the situated alignment for the exp approach, which is computed as follows. First, whenever the agent is in state q and receives v_2, it initializes the value of the possible mappings for v_2 that are not yet in $\omega_{\texttt{exp}}$. For all $v_1 \in U_1(q)$:

$$\omega_{\texttt{exp}}(q, v_1, v_2) := \begin{cases} 0 & \text{if } \langle q, v_1, v_2 \rangle \in dom(\omega_{\texttt{exp}}) \\ \omega_{\texttt{exp}}(q, v_1, v_2) & \text{otherwise} \end{cases}$$

Recall that agents maintain a sequence $(q^1, {v_1}^1, {v_2}^1), \ldots, (q^n, {v_1}^n, {v_2}^n)$. When the interaction ends, the agent updates the values of each (q, v_1, v_2) in the sequence of mappings made according to whether the interaction was successful or not:

$$\omega_{\texttt{exp}}(q, v_1, v_2) := \begin{cases} \omega_{\texttt{exp}}(q, v_1, v_2) + 1 & \text{if the interaction succeeded} \\ \omega_{\texttt{exp}}(q, v_1, v_2) & \text{if the interaction failed} \end{cases}$$

Now, let us explain how the situated alignment $\omega_{\texttt{exp}}$ is used to choose interpretations when agents receive a foreign message. Suppose a_1 receives a word v_2 in a state q. In the experience approach, it will choose an interpretation $v_1 \in V_1$ with a probability proportional to a value $p_{\texttt{exp}}(q, v_1, v_2)$, that is computed as follows. Let ξ be a parameter with value close to 1, and $\#exp(q, v_2) = \sum_{{v_1}' \in U_1(q)} \omega_{\texttt{exp}}(q, v_1', v_2)$. That is, $\#exp(q, v_2)$ is positive if there is a mapping for v_2 in q that was part of a successful interaction before, and 0 if there is none.

$$p_{\texttt{exp}}(q, v_1, v_2) = \begin{cases} \xi \dfrac{\omega_{\texttt{exp}}(q, v_1, v_2)}{\#exp(q, v_2)} + (1 - \xi) \frac{1}{|U_1(q)|} & \text{if } \#exp(q, v_2) > 0 \\ \dfrac{1}{\mid U_1(q) \mid} & \text{if } \#exp(q, v_2) = 0 \end{cases}$$

This method divides the matching decisions into two phases. When there is not enough information from the experience, because no mapping was correct,

the agent maps randomly. Once interactions start to be successful, it repeats good choices, favoring those that have been successful in many cases. The exploration parameter ξ is included to consider situations in which the configuration of the interaction protocols can make two mappings be correct in one state.

Note that agents learn only from successful interactions. Since the outcome of the interaction is only known once it finished, this mechanism is affected by the *delayed reward* problem: learning from unsuccessful interactions is difficult, because it is not known which of the mappings in the sequence was wrong.

The Experience Approach in Action.

Recall the Travel Agency example depicted in Figure 3.1, where a travel agent (TA) interacts with a customer (C). Suppose $SP = \{success, failure, book, info\}$ and the following state property function for both interaction models: $\rho(7) = \{success, book\}$, $\rho(8) = \{failure, book\}$, $\rho(11) = \{success, info\}$.

The interaction model IM_C for the customer is compatible with IM_{TA}. Let T be the pragmatic alignment under which these are compatible. An example of a successful interaction between these interaction models is the sequence

$$\langle \mathsf{Hotel}, \mathsf{Accommodation} \rangle, \langle \mathsf{hotelBookIn}, \mathsf{city} \rangle, \langle \mathsf{show}, \mathsf{show} \rangle$$

This implies, for example, that in the pragmatic translation from IM_{TA} to IM_C, $\langle 9, \mathsf{hotelBookIn}, \mathsf{city} \rangle \in T$.

Using approach `exp`, the travel agent will first go through a learning phase, in which the interactions will be mostly unsuccessful since it is choosing mappings randomly. However, since the interaction has few interpretations choices and each of them with few options, it should not take long to find correct mappings. We show this experimentally in Section 3.5.

3.4 Using External Vocabulary Alignments

A different approach to find a translation between the vocabularies of agents that need to interact consists in using aligning external tools to the interaction, taking advantage of the multiple matching techniques that have been developed in the last decades. These techniques vary from sophisticated matchers of logic-based ontologies to syntactic similarity measures. The choice between the existent possibilities depends on the type of additional semantic structure that comes along with the vocabularies of the agents, the possibility of accessing this structure, and the availability of the matching tools. For our purposes we can consider all these options in the same way, focusing directly on the *alignment* between vocabularies that they provide, which is defined as follows.

Definition 3.8. We will call *mapping* a pair $\langle v_1, v_2 \rangle$ of a local word $v_1 \in V_1$ and a foreign one $v_2 \in V_2$. An *alignment* between two vocabularies V_1 and V_2 is a partial function $\alpha : V_1 \times V_2 \to [0, 1]$, that relates mappings with a confidence value.

The function is partial because not all mappings are necessarily defined. Let $v \in V_2$. We will call $dom_v(\alpha)$ the words $v_1 \in V_1$ such that $\langle v_1, v \rangle \in dom(\alpha)$ is defined. The same can be defined for words in V_1. ■

We consider a pair of words in $dom(\alpha)$ to be aligned if they are assigned a confidence higher than 0. If a mapping has value 0 or is not in $dom(\alpha)$, we consider it to be not aligned.

The definition of alignment that we propose differs in some ways from the one commonly used in the ontology matching literature (Bouquet et al. 2004). First, alignments are generally relations (that is, a set of tuples with two words and a confidence) instead of functions. We choose a functional formulation because it will be useful to explain operations over alignments later, and to restrict ourselves automatically to having only one confidence per mapping. More importantly, alignments in general also include a *relation type* that determines how the two words are related. Commonly, the types can be *equivalence* or *subsumption*. The latter is a typical notion in hierarchies, and is used to relate two concepts, one of which is more general than the other, like for example Beer and Drink. In this work we will only work with equivalence, and it is left for future work to investigate how subsumption could be included.

The quality of vocabulary alignments is typically measured by comparison to a *reference alignment* that is considered to contain all the correct mappings. The alignment is commonly compared with this reference using the *precision* and *recall* measures which compute, respectively, how many of the mappings in the alignment to evaluate are correct according to the reference, and how many of the mappings in the reference were found. To use the standard definitions, we need to use our alignments as relations. We use the simplest definition of precision and recall, which does not take into account the confidence degrees in these measures (see (Euzenat and Shvaiko 2013) for this and alternative methods). Consider an alignment α and a reference α^r between V_1 and V_2.

Definition 3.9. The *precision* of an alignment α with respect to a reference alignment α^r is the fraction of the mappings in α that are also in α^r:

$$precision(\alpha, \alpha^r) = \frac{\mid dom(\alpha) \cap dom(\alpha^r) \mid}{\mid dom(\alpha) \mid}$$

while its *recall* is the fraction of the mappings in α^r that were found by α:

$$recall(\alpha, \alpha^r) = \frac{\mid dom(\alpha) \cap dom(\alpha^r) \mid}{\mid dom(\alpha^r) \mid}$$

■

We said that we would consider only mappings with values greater than 0 as aligned. To this end, the precision and recall of α are obtained by computing those values for another alignment α^+, which only contains those mappings in α that have value greater than 0.

Let us describe in detail how an external alignment can be used to let semantically heterogeneous agents communicate. Again, we will describe the process from the point of view of a_1, but it is analogous for its interlocutor. We consider the communicative situation described in the previous section, where two agents a_1 and a_2 want to communicate, but they speak different vocabularies V_1 and V_2 respectively. We assume a_1 has a previous alignment $\alpha_1 : V_1 \times V_2' \to [0, 1]$. Note that the alignment is not necessarily defined over V_2; this is because a_1 may not know the foreign vocabulary a priori, and the alignment may include mappings with words that a_2 does not actually use. As long as $V_2 \cap V_2' \neq \varnothing$, the alignment can be useful. The most straightforward approach to use α_1 is to apply it directly for translating words in messages. If a_1 receives v_2 while waiting for words in a set, it will choose the word that matches with v_2 with the highest confidence in the alignment; if there is no such word, it chooses one randomly.

This approach will work well if the alignment α_1 is adequate for the task the agents are performing. Low recall means the agent has to make more choices randomly, which can cause unsuccessful interactions. Low precision implies a higher probability of choosing incorrect matches, which can also cause an interaction to fail.

Alignment Approach (`align`).

As before, agents have a situated alignment, which now depends only on the information in the previous alignment. In this case, there is no learning process. The first time a_1 receives v_2 in q, it initializes the situated alignment as follows for each $v_1 \in U_1(q)$:

$$\omega_{\texttt{align}}(q, v_1, v_2) := \begin{cases} \omega_{\texttt{align}}(q, v_1, v_2) & \text{if } v_1 \in dom_{v_2}(\alpha_1) \\ 0 & \text{otherwise} \end{cases}$$

One way of mitigating the effect of low recall is to consider not only the mappings that are explicitly present in the alignment, but to take into account the additional structure that (local) vocabularies may have. For example, if agents have a similarity measure between their words, they can consider not only those mappings in the alignment, but also words that are similar to the ones mapped. We do not formalize this idea here. To reduce the effects of low precision, a solution is to not trust the alignment completely, including

an exploration parameter ξ' in the definition of a probability distribution over words in $U_1(q)$.

Once the situated alignment is defined, it only remains to describe how it is used to choose interpretations for foreign words. If a_1 receives v_2 in state q, let $\hat{\omega}_{align}(v_1, v_2)$ be the normalized value of $\omega_{\texttt{align}}(v_1, v_2)$ for each $v_1 \in U_1(q)$. Then the agent chooses $v_1 \in U_1(q)$ with probability:

$$p_{\texttt{align}}(v_2, v_1) = \xi' \times \hat{\omega}_{\texttt{a}align}(v_1, v_2) + (1 - \xi')\frac{1}{\mid U_1(q) \mid}$$

Choosing the value for the exploration parameter ξ' introduces a dilemma related with how strongly should the alignment be trusted. Using large values results in a very deficient method when there are wrong mappings in the alignment, while with low values we can be losing useful information. Ideally, the parameter should depend on the precision of the alignment α_1, but this is something agents are not expected to know in advance.

The Alignment Approach in Action: A Running Example

In what follows we introduce an illustrative example using a portion of a mapping between the travel agent's ontology and the one of the customer. The complete vocabularies and the specifications of the ontologies used by the agents can be found in (Atencia 2010); we do not need them explicitly here.

$v_1 \in V_1$	$v_2 \in V_2$	Confidence
Return	Package	0.41
Single	RoundTrip	0.19
UnregCustomer	OneWay	0.03
Flight	Customer	0.01
destination	airlineCompany	0.99
carrier	to	0.99
departing	leavingDate	0.99
origin	from	0.99
returning	returnDate	0.76
hotelBookingsIn	city	0.30

Table 3.1: Extract of the alignment obtained with the ontology matcher Falcon-AO as reported by Atencia (2010)

Consider again the travel agent TA and customer C interacting with the models in Figure 3.1. Now, to be able to interact with C, agent TA uses an alignment provided by an external source. Table 3.1 shows a relevant fragment of the alignment provided by the matcher Falcon-AO (Jian et al. 2005)

as reported in (Atencia 2010). Consider a situation in which TA is in state 4, where $U_{TA}(4) = \{OneWay, RoundTrip\}$. Using the alignment criteria, if C sends Single, the travel agent will probably interpret it as RoundTrip, since $\omega_{\texttt{align}}(4, \mathsf{Single}, \mathsf{RoundTrip}) = 0.19$ while $\omega_{\texttt{align}}\ (4, \mathsf{Single}, \mathsf{OneWay}) = 0$.

3.5 A Combined Technique

In this section we will discuss how to combine the two approaches that we introduced in the last section. That is, how to integrate an external alignment with the method that learns a mapping from the experience of interacting with compatible protocols. We propose an integration that keeps in mind that the ultimately correct alignment is one that allows agents to interact successfully. We consider again a_1 interacting repeatedly with a_2; now, in addition, a_1 has access to an external alignment α_1 between V_1 and V_2.

Since the alignment α_1 was produced by external resources, it does not necessarily agree with the pragmatic translation T between IM_1 and IM_2. There is a situation in which a wrong mapping in an external alignment can be particularly harmful for the interaction. The problem arises when $(v_1, v_2) \in dom(\alpha_1)$ and $v_1 \in U_1(q)$, but (q, v_1, v_2) does not belong to the pragmatic translation between the interaction models of both agents. In the travel agency example, and with α as in Table 3.1, this happens in state 4, because, from the point of view of the travel agent, $(\mathsf{RoundTrip}, \mathsf{Single}) \in dom(\alpha_1)$ and $\mathsf{RoundTrip} \in U_1(4)$, but $(4, \mathsf{RoundTrip}, \mathsf{Single}) \notin T$ because it does not lead to any successful interaction. When the alignment is followed, most of the times RoundTrip will be chosen as an interpretation for Single, causing the interaction to fail. We will refer to this kind of mappings as *misleading*. In Section 3.7 we will discuss how the quality of an alignment can be measured with respect to a pair of interaction protocols. Now, we present two different ways of integrating previous alignments, solving the problem of misleading mappings.

The first combination that we propose is simple. The matching process is again divided in two phases. In the first one, instead of choosing randomly, agents take the external alignment into account. Once they obtain information from successful interactions they continue as in `exp`, preferring those mappings that were successful.

Alignment and Experience Approach (`align-exp`).

In this direct combination, the agent computes $\omega_{\texttt{exp}}$ and $\omega_{\texttt{alg}}$ as before, updating the values of the former one when it receives a new message.

These values are used to choose an interpretation as follows. Consider agent a_1 who receives a foreign message v_2 in state q. Recall that $\#exp(q, v_2) =$

$\sum_{v_1{}' \in U_1(q)} \omega_{\texttt{exp}}(q, v_1', v_2)$ is positive only if there has been a successful mapping for v_2 before. The probability that the agent chooses an interpretation $v_1 \in U_1(q)$ for v_2 is computed as follows:[1]

$$p_{\texttt{align-exp}}(q, v_1, v_2) = \begin{cases} p_{\texttt{exp}}(q, v_1, v_2) & \text{if } \#exp(q, v_2) > 0 \\ p_{\texttt{align}}(q, v_1, v_2) & \text{if } \#exp(q, v_2) = 0 \end{cases}$$

This straightforward combination shares some of the problems of the individual methods it builds on. First, it still considers only the successful matches and discards all the information in the ones that failed, since it does not have a way of deciding which mappings to punish. Second, the dilemma of when to choose randomly instead of following the alignment that we explained in the alignment approach `align` is not solved.

3.5.1 Learning from Unsuccessful Experiences

Until now, the methods we described use only information obtained from successful interactions. However, succeeding is difficult, since agents have to make the correct interpretating choices in all states. If protocols are large, this could be very unlikely. As we already mentioned, what hinders the use of unsuccessful interactions is known as the problem of *delayed reward*. Concretely, our agents do not receive any feedback on the interpretations they make until the interaction finishes; successfully or not. When the interaction is successful this is not a problem, since all the mappings are, by definition, correct. However, when the interaction fails the agents have no information about which interpretation they chose mistakenly. Even more, they do not know whether themselves or their interlocutor made a mistake. The best they can do in this situation is to punish all the mappings they made. However, this is dangerous. Only one wrong mapping is enough to fail, and punishing correct ones can be very harmful.

We now present a more sophisticated method that takes into account the existent alignment in a way that allows agents to also learn from successful interactions. To avoid punishing correct mappings, this method tries to find out when the first wrong interpretation was made. To this aim, the confidence in a particular mapping is updated taking into account not only the final result of the interaction, but also the quality of the mapping possibilities that were found subsequently. The intuition behind this idea is that, if good mappings were found after a particular choice of interpretation, it is likely that it was correct. Consider a simple analogy with human conversations: if someone is not sure of having understood a message, but the dialog continues as expected,

[1] It would be reasonable to use $p_{\texttt{align}}(q, v_1, v_2)$ in the exploration of `exp` instead of following a uniform distribution, we do not add it for clarity. The same holds for the next approach.

she will assume her understanding was correct, whereas if strange messages arrive, her confidence in her previous understanding will decrease. This technique deals with low quality alignments in a smarter way. It mitigates low precision by punishing mappings in unsuccessful interactions instead of resorting to the exploration parameter. And it mitigates low recall by also considering, for a given mapping, the information about other mappings in the interaction.

Our method, again, divides the learning into two phases. The difference with the already presented criteria is in the first phase. Before, this phase consisted in choosing a word according to a uniform distribution, or to an alignment. Now, agents also compute a distribution over the possible interpretations in this phase, that represents their confidence in a mapping belonging to the pragmatic translation. This value depends on the external alignment and on information obtained from unsuccessful alignments.

In this first phase, we use a method that resembles classical temporal difference reinforcement learning techniques (Sutton and Barto 1998). These methods are often used to predict the reward obtained from each action, when the predictions in one state are considered to be related to subsequent ones. In our case, we consider the reward to be the likelihood of possible mappings. That is, agents receive a large reward in a state if there is a word in the set of expected messages that maps with the received one with large confidence. In this way, we use the certainty provided by the alignment as a reward. In addition, agents receive a punishment when an interaction ends in failure. This new distribution is also updated when the interaction finishes, taking into account the rewards and punishments observed. The initial values in this distribution are computed with the ones from the external alignment.

Evolving Alignment and Experience Approach (`evol`).

We will call $\omega_{\texttt{evol}}$ the situated alignment that agents use while they still have no information about successful interactions. This alignment is maintained together with $\omega_{\texttt{exp}}$, which is still updated and used to know which words were mapped in a successful interaction.

The situated alignment $\omega_{\texttt{evol}}$ is updated as follows. First, whenever the agent receives v_2 in q, it initializes it using the values in the external alignment α_1. For each $v_1 \in U(q)$,

$$\omega_{\texttt{evol}}(q, v_1, v_2) = \omega_{\texttt{align}}(q, v_1, v_2)$$

When an interaction finishes in failure, the agent has a sequence (q^1, v_1^1, v_2^1), $\ldots$, (q^n, v_1^n, v_2^n) of the mappings it made, like in the `exp` method. We assume the sequence is ordered in the same way the mappings were made. In this case, the agent made n mappings. For each of these states, the agent updates

$\omega_{\texttt{evol}}(q, v_1, v_2)$ as follows. Let $\sigma \in (0, 1]$ be a *forgetting parameter*, and $\theta \in (0, 1]$ a *punishment*.

- If $i = n$, a punishment of $-\theta$ is assigned for having failed:

$$\omega_{\texttt{evol}}(q^n, v_1^n, v_2^n) := (1 - \sigma)\omega_{\texttt{evol}}(q^n, v_1^n, v_2^n) + \sigma(-\theta)$$

- For $0 < i < n$, the agent takes into account the mapping possibilities it encountered in the subsequent states. The forgetting parameter is used to make values be more important to closer mappings than to those far away:

$$\omega_{\texttt{evol}}(q^i, {v_1}^i, {v_2}^i) := (1 - \sigma)\,\omega_{\texttt{evol}}(q^i, v_1^i, v_2^i) + \sigma \max_{v \in U_1(q_{i+1})} \omega_{\texttt{evol}}(q^{i+1}, v^{i+1}, v_2^{i+1})$$

When a message is received, an interpretation is chosen as follows. Let $max = \text{argmax}_{v \in U_1(q)}(\omega_{\texttt{evol}}(q, v, v_2))$ and, again, $\#exp(q, v_2) = \sum_{{v_1}' \in U_1(q)} \omega_{\texttt{exp}}(q, v_1', v_2)$. Choose $v_1 \in U_1(q)$ with probability:

$$p_{\texttt{evol}}(q, v_1, v_2) = \begin{cases} p_{\texttt{exp}}(q, v_1, v_2) & \text{if } \#exp(q, v_2) > 0 \\ \frac{1}{|max|} & \text{if } \#exp(q, v_2) = 0 \text{ and } v_1 \in max \\ 0 & \text{if } n = 0,\ v_1 \notin max \end{cases}$$

Now we do not need to allow explicitly for exploration, since the back-propagation of the punishment already has that effect. For this reason, the value is 0 when the word is not between the interpretations with higher value. Note that, since values are updated when the interaction is over, the new maximum value for future states can be used. This will back-propagate the punishment to all mappings already in the first unsuccessful interaction, repairing misleading mappings in less interactions, although it is less stable. This is the approach we use in the experimentation in Section 3.6.

We now provide an intuition of the mechanism used by `evol` to repair misleading mappings. Consider a misleading mapping between v_1 and v_2 in q and all mappings made after that one in an interaction. If none of these were positive mappings in the external alignment α_1, the value of $\omega_{\texttt{evol}}(q, v_1, v_2)$ will decrease. This may not be enough to make it lower than other options, but since the values of subsequent mappings will never increase, $\omega_{\texttt{evol}}(q, v_1, v_2)$ will continue to decrease. If, on the other hand, there are positive mappings, they have to be misleading, so they will also be repaired eventually, arriving eventually to the situation when there are no positive mappings. It can be the case that these mappings are correct for other strings, but since correct mappings do not modify the values, they will not damage the process unless the case above occurs, preventing one mapping of being chosen. Since this is true for any interaction that includes the mapping (q, v_1, v_2), it will eventually be repaired.

Approach	`align`	`exp`	`align-exp`	`evol`
successes (%)	30	92	81	96
convergence	-	7.1	20.1	3.9

Table 3.2: Results for the Travel Agency scenario, for a total of 60 experiments, where *convergence* is after how many interactions agents started to interact always successfully.

3.5.2 The Combination Methods in Action

Let us analyze the performance of the two criteria presented in this section when applied to the travel agency scenario. As we already mentioned, there is a misleading mapping between Single and RoundTrip; as a consequence, the `exp-alg` method will fail at least until the agent chooses to explore. This is solved in few interactions when using the `evol` approach. Since the interaction fails right after Single is mapped with RoundTrip, the fourth approach will find and solve this error in just one unsuccessful interaction. Also the first time after choosing it, the agent can confirm the (Flight, Flight) mapping, since its confidence will be increased with the (leavingDate, departing) and (from, origin) mappings.

To evaluate our predictions, we studied the performance of the four criteria in the Travel Agency scenario experimentally, letting agents interact 60 times. We measured the proportion of successful experiences, as well as after how many interactions they always understood each other. The results, which are as expected, are shown in Table 3.2. The approaches `align` and `align-exp` are worse than `exp`, since they rely on the alignment and are hindered by the misleading alignment. The approach `evol`, which uses the alignment intelligently, converges much faster than any other one.

3.6 Experimental Evaluation

We evaluated the methods that we propose experimentally, by observing how they perform when used by heterogeneous agents. Our objective is to determine 1) if the methods help agents interact successfully, and 2) which methods work best in different situations. Before presenting the results, we discuss the generation of data for experimentation.

3.6.1 Data Generation

The first challenge faced when designing the experimentation is related to the test data. The interaction models that we defined are abstract formalizations,

that are useful to define a general technique, but difficult to find as a real-world dataset with enough examples to analyze our methods with. Even more if we needed them to be in two languages. While it is possible to adapt the techniques to be useful for more concrete protocols, we wanted to evaluate the general method. For these reasons, the most reasonable option was to generate the test data randomly.

While it is simple to build random finite state automata, it is not clear that all possible protocols model a realistic interaction, and the literature does not offer a useful characterization of interaction or conversation protocols. We chose to generate deterministic automata parametrized by their size (given by the number of states) and to only restrict their shape by using a uniform distribution of the outgoing arrows among the states.

Then, we created vocabularies V_1 and V_2 randomly and defined a bijective translation between them. We labeled an interaction protocol IM_1 with words in V_1 and a compatible one IM_2 with its translations to V_2, generating in this way the correct pragmatic translations that allowed agents to interact successfully. Finally, we created alignments between V_1 and V_2 of different quality with parametrized values of precision and recall with respect to the pragmatic translation. To compute the precision and recall with respect to a pragmatic translation, we simply considered the relation without the states, removing duplicates. We used confidences of 1 for all the relations in the generated alignments. Note that, although the method is defined more generally, we only conducted experiments with bijective alignments. However, this should not affect the performance of the techniques, since agents consider only alignments parametrized by states.

3.6.2 Experiments

The performance of the methods we propose can be analyzed in at least three different dimensions:

1. **The complexity of the interaction models.** One possibility is to analyze how the techniques work for protocols of different sizes or shapes. We decided not to focus on this dimension, mainly because it had been already explored by Atencia and Schorlemmer (2012). We used protocols with a fixed size of 100 transitions and between 50 and 90 states, and a vocabulary of 50 words for all experiments.

2. **The value of the parameters.** For `evol`, we experimented with different values of σ and θ, concluding that low (between 0.2 and 0.4) values of σ gave the best results. The results for the punishment were less clear, but values between 0.7 and 0.9 seemed to be better. A hypothesis that should be confirmed is that this depends on the average of the mapping

confidences in the alignment. We used $\sigma = 0.3, \theta = 0.8$, and $\xi = \xi' = 0.9$ for the exploration parameters in the alignment and experience criteria.

3. **The quality of the alignments.** This dimension analyzes how our methods perform for alignments that have different values of precision and recall with respect to the pragmatic translation. This dimension turned out to be the most interesting one, and we develop it in detail in this section.

Experiment 1: General Performance

The first experiment we performed provides a general comparison of the four methods. A run of this experiment is composed of two agents that use the same matching approach, following compatible protocols IM_1 and IM_2, each of them with an alignment with given values of precision and recall. For each of the four approaches, we let agents go through a learning phase in which they interacted n times, running the experiment for $n = i^2$ and $i \in [2, 20]$. After this training phase, we let agents interact again 100 times, without knowledge update, and measured the proportion of successful interactions. We performed 50 repetitions of each run, each time with a different alignment (and a different number of states), but maintaining the same values of precision and recall.

We considered three quality classes for the precision and recall values: `low`: 0.2, `medium`: 0.5, `high`: 0.8, and evaluated the approaches that use an alignment with the nine resulting combinations. Figure 3.2 presents the obtained results, showing the proportion of successful interactions for different lengths of the training phase. We only show eight of the nine cases, but the remaining one follows the same trend. Repeated interactions have no effect on the `align` alignment; we plot the result of one experiment as a constant. The same happens for `exp` with different alignment qualities.

The two methods that combine the alignment and the learning from the experience perform better in general. Between them, `evol` is always the best one, performing better than all other methods. This method achieves 90% of correct matches after only $\sim$ 60 interactions. The method that only uses the learning also reaches values of success close to 1, but more slowly. Using only the alignment is the worst option, except after very short training periods. A more detailed analysis provides interesting observations about precision and recall:

- Recall affects performance more drastically than precision. This becomes clear when comparing the success rate for `align`; while it increases significantly with higher values of recall, there is much less variation with different precision values. The two combined methods are also much better with high recall. This shows that errors in a contextualized environ-

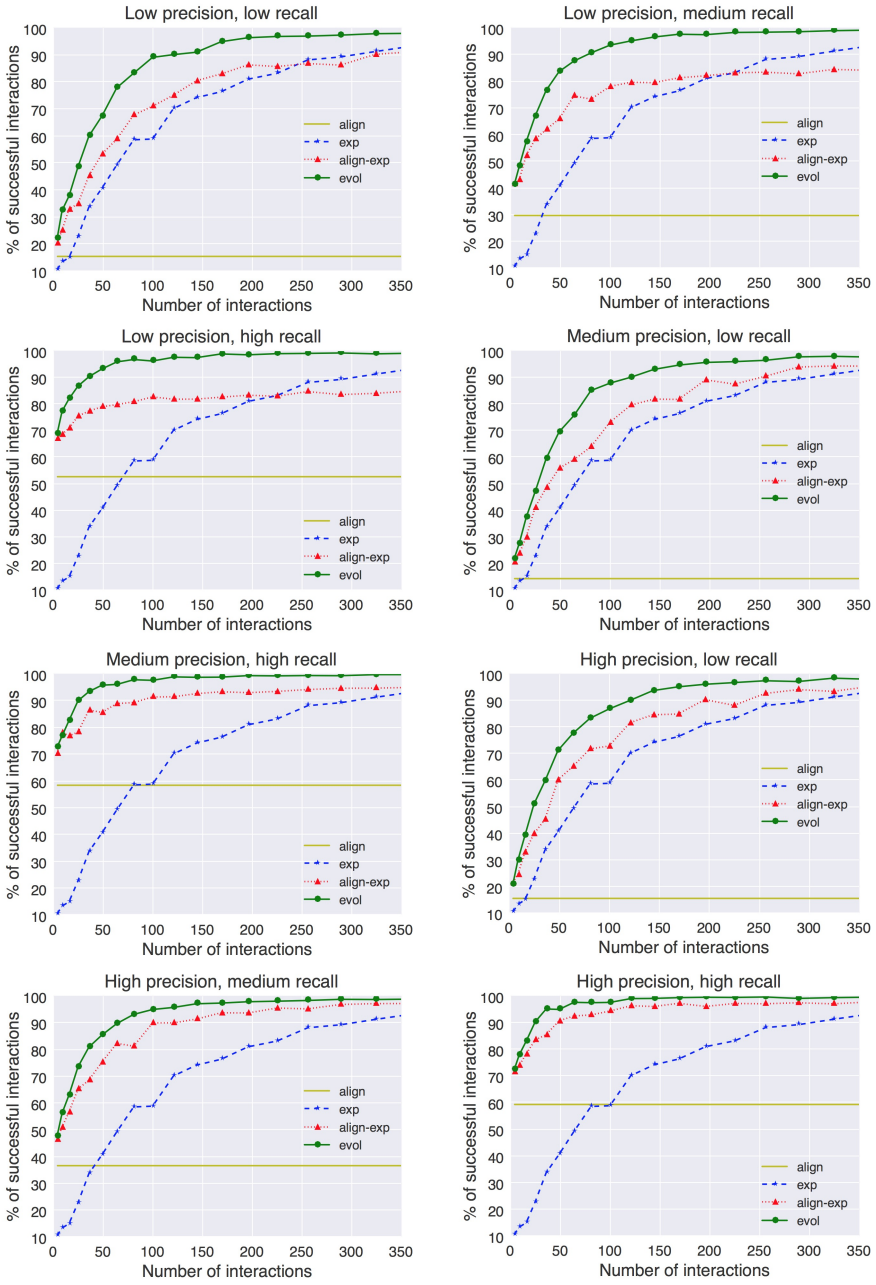


Figure 3.2: Results for Experiment 1, with size=90 and different alignment qualities

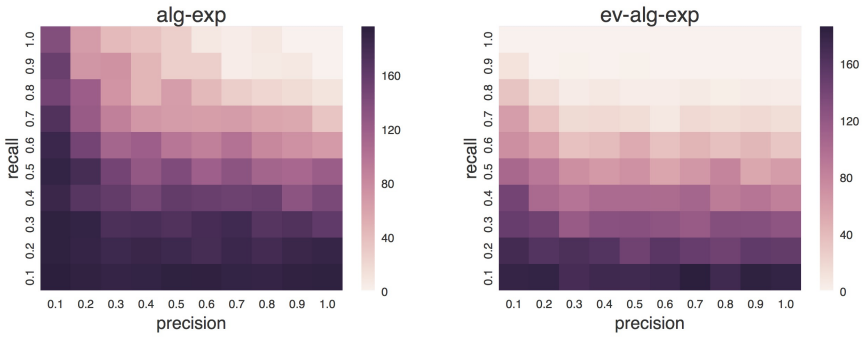


Figure 3.3: Results for Experiment 2

ment are less dramatic, because it is more rare to find one in the expected messages.

- With low levels of precision, `evol` is significantly better than `align-exp` after longer learning phases. This can be seen in the plots for low precision, particularly for high or medium recall, where the evolutionary technique reaches values close to 1 while `align-exp` does not, being even worse than the technique without the alignment. This is explained because low precision implies higher possibility of misleading matches, which are only solved by making the alignment evolve.

- With low levels of recall, `evol` learns faster after short training periods. This is because it takes into account good future mappings, using the available information more efficiently.

Experiment 2: Focus on Precision and Recall

In Experiment 1, the effects of using alignments of different qualities are only hinted at. To analyze in depth how the performance of our techniques changes with different values of precision and recall, we developed a second experiment. In Experiment 2, we let agents interact a large number of times (fixed as 200) and measured after how many interactions they found a good alignment when using the techniques `align-exp` and `evol`. In this case we considered agents to converge when they had 90% of successful interactions in the following interactions. The results are shown in Figure 3.3. The color gradient represents the number of interactions before convergence, which increases with darkness. For the `align-exp` technique, both low precision and low recall affect the performance, only converging fast when both values are high. As we already pointed

out, low recall is more harmful than low precision. In the results for `evol` it can be seen that the precision has less influence; with high levels of recall, low values of convergence are achieved even with very low precision. This again shows how letting the alignment evolve repairs misleading mappings, solving low quality in this dimension.

3.7 A Pragmatic Approach to Alignment Evaluation

In the previous sections we discussed how pre-existing alignments can be combined with shared knowledge of how to perform a task. The objective of this combination is twofold: it makes the convergence to a useful alignment faster when the external mappings are good, and it completes and repairs the alignment when it is not adequate for that particular interaction. In doing so, we discovered that the quality of an alignment for a particular task depends not only on the mappings that belong to its pragmatic translation, but also on the shape of the interaction models. Concretely, we talked about *misleading* mappings, which are particularly problematic. In this section we address the question of how the act of interacting can be used to determine the quality of an alignment, taking these insights into account. Our objective here is not to obtain a better alignment but to evaluate the ones that are produced by an external tool. We propose an application-dependent evaluation technique that does not require the (possibly idiosyncratic) construction of a gold standard. In this way, we make a step towards considering the problem of "in situ evaluation", based on the idea that "the relative quality or usefulness of a generated alignment also depends on its intended use" (Euzenat et al. 2011).

Normally, the quality of a vocabulary alignment is measured in comparison with a *reference alignment*, using the *precision* and *recall* measures that we defined in Section 3.4. These notions were originally developed for the problem of information retrieval, and their use for the evaluation of ontology alignments has been called into question by different authors. These criticisms in general argue that these measures overlook important aspects of the problem that should also be taken into account to decide how good a solution is. For this reason, some authors have proposed measures that are more appropriated for the nature of semantic mappings. One example is the idea of *semantic precision and recall* (Euzenat 2007). Here, Euzenat tackles the problem of the binary nature of traditional precision and recall (if a mapping is not found by the alignment, it is considered to be not aligned), by considering the relation between the logical consequences of the alignments instead of between the alignments themselves. Hollink et al. (2008) propose new evaluation measures that take into account the frequency of use of the mappings found, as well as the semantic distance to an alignment. van Hage et al. (2008) introduce the notion of *relevance* of

a mapping, that measures how often the mapped words appear in a particular context.

When alignments are used to facilitate the interaction between agents that speak different languages, the standard precision and recall measures face two main problems. First, as with other applications, it is possible that no reference alignment between the vocabularies is available. This can happen, for example, if the vocabulary that is used is very specific to the interaction being performed. Second, the measures do not take into account the way in which terms are used in an interaction. In Section 3.5 we found out that the recall of the alignment is more important than its precision when it is used to interpret utterances. This is because agents are expecting a reduced set of words, so it is unlikely that one of them will be incorrectly mapped with the received word.

3.7.1 Pragmatic Precision and Recall

In this section we present a new definition of precision and recall, specially designed to evaluate alignments that are going to be used by interacting agents. We consider two vocabularies V_1 and V_2 and the already defined interaction models IM_1 and IM_2 over them. Concretely, we classify mappings as those that are helpful to finish an interaction successfully and those that are particularly harmful. This leads to the notion of *useful* and *misleading* mappings, which are, respectively, those that lead to the success or failure of an interaction. We have already introduced misleading mappings in Section 3.5, identifying them as particularly problematic; here they are defined formally. This new classification allows us to compare an alignment against the specification of an interaction, providing a method for evaluating alignments that does not rely on a human-crafted alignment. We then show how these newly defined measures can be used by agents to improve their mutual understanding, and sketch a method by which agents can estimate them dynamically using their experience from interaction, obtaining an unsupervised evaluation method.

Definition 3.10. Let α be an alignment between vocabularies V_1 and V_2 and recall interaction models IM_1 and IM_2. We say a mapping $\langle v_1, v_2\rangle \in dom(\alpha)$ is *useful* with respect to IM_1, IM_2 if $\langle v_1, v_2\rangle$ appears in a successful interaction in the communication product $IM_1 \otimes IM_2$. The mapping $\langle v_1, v_2\rangle$ is *misleading* if it appears in an unsuccessful interaction in $IM_1 \otimes IM_2$. ■

Notice that there can be mappings in $dom(\alpha)$ that are neither useful or misleading. We will call *relevant* the mappings that can be classified in one of these categories, or equivalently, mappings between pairs that belong to an interaction in the communication product between the models. More surprisingly, a mapping can be both useful and misleading at the same time, if they correspond to

different states. This allows for different possibilities when computing precision and recall. Here we consider as correct all useful alignments.

To define precision and recall for α with respect to IM_1 and IM_2, let *useful* and *relevant* be, respectively, the sets of useful and relevant mappings of α with respect to the interaction models. Let T be the pragmatic translation under which IM_1 and IM_2 are compatible, and let us define $pragmatic = \{\langle v_1, v_2\rangle \mid \langle q, v_1, v_2\rangle \in T \text{ for some } q \in Q\}$. Pragmatic precision and recall are defined as follows:

$$recall^{pr} = \frac{\mid useful \mid}{\mid pragmatic \mid}$$

$$precision^{pr} = \frac{\mid useful \mid}{\mid relevant \mid}$$

As argued in (Hollink et al. 2008), we may want to take into account not only how many, but also which of the mappings are found by the alignment. Finding a correct mapping for a very common word should have more impact on precision than finding a mapping for a rarely used one. This can be taken into account in the pragmatic precision and recall measures we just defined, by simply considering *useful* and *relevant* as multi-sets:

- *useful:* for each state $q \in Q$, all mappings in T that are useful in q
- *relevant:* for each state $q \in Q$, all mappings in T that are relevant in q

Precision is defined in the same way, and recall as:

$$recall = \frac{\mid useful \mid}{\mid T \mid}$$

Note that, with these definitions, it may be impossible to obtain alignments with certain value of precision and recall. The values will be determined by the structure of interaction models. For example, consider a linear interaction model in which each state has only one outgoing arrow. There are no possible misleading matches with this protocol; therefore the minimum level of precision for alignments is necessarily 1.

Example 2: Ordering Drinks

In Figure 3.4 we present the interaction models for the waiter and customer of the Ordering Drinks interaction that we presented before. In a nutshell, the customer can ask for wine or beer, and according to this choice the waiter will ask for more details.

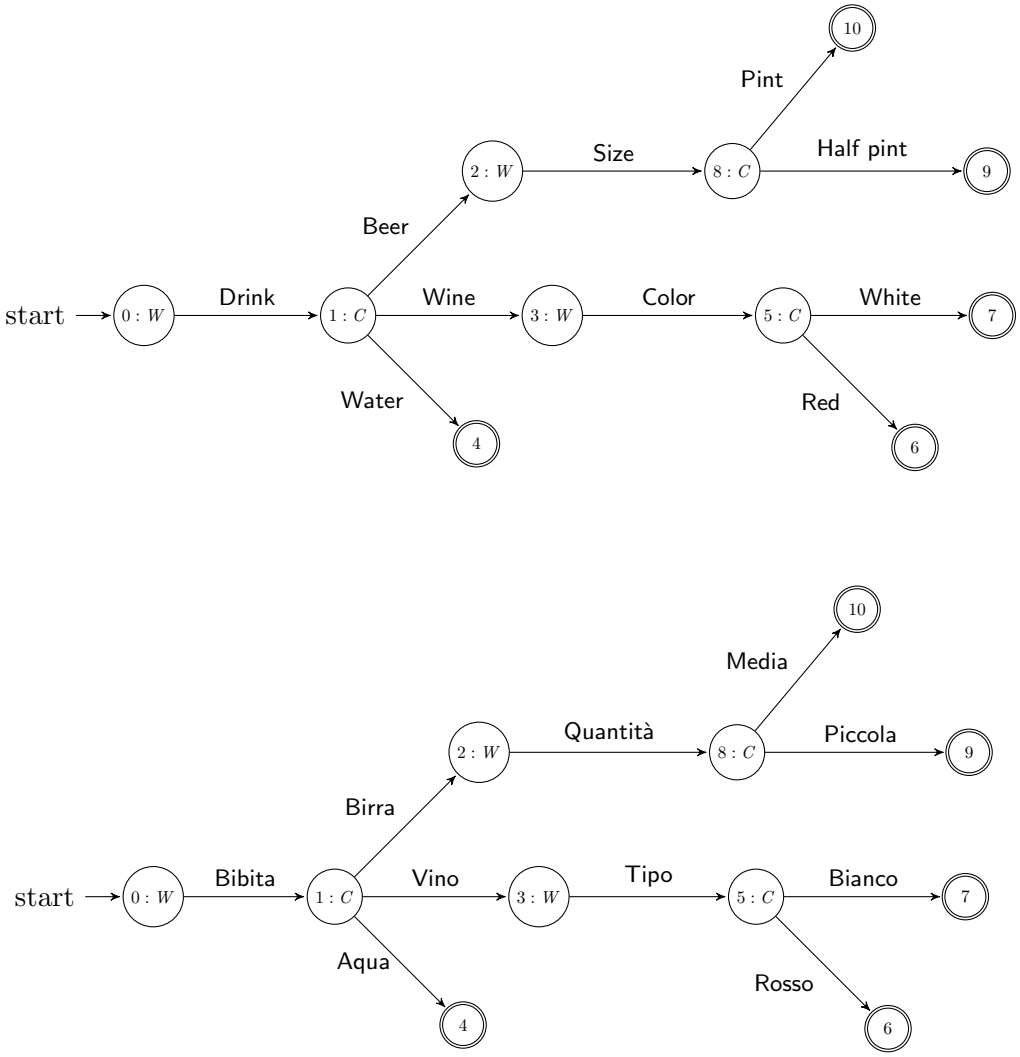


Figure 3.4: English and Italian interaction models for ordering drinks

Alignment 1		Alignment 2	
$v_1 \in V_1$	$v_2 \in V_2$	$v_1 \in V_1$	$v_2 \in V_2$
Bibita	Water	Bibita	Water
Vino	Wine	Vino	Wine
Rosso	Red	Rosso	Red
Quantità	Pint	Media	Half Pint

Table 3.3: Two alignments for the *ordering drinks* example.

For this example, consider the alignments in Table 3.3. According to an English-Italian dictionary, both mappings would have precision 0.5 ((Wine, Vino) and (Red, Rosso) are correct). Since Media means Half in Italian, the mapping (Media, Half Pint) could also be considered correct, giving the second alignment a precision of 0.75. However, these alignments are clearly not equally useful when used by agents that are interacting because the second alignment has a misleading mapping (Media, Half Pint). Using our values, both alignments have a recall of 0.2 ((Wine, Vino), (Red, Rosso) are the useful alignments found), but the first one has a precision of 1 and the second one of 0.66.

3.7.2 Pragmatic Precision and Recall in Practice

In their pragmatic version, precision and recall are not only indicators of how useful an alignment is for a particular interaction, but can also be used actively by semantically heterogeneous agents to improve their mutual understanding. In this section we focus on the practical application of the evaluation of pragmatic translations. We first analyze how pragmatic precision can be used to improve the use of the alignment in interaction, and then sketch a method in which agents can estimate their values by interacting.

Using Pragmatic Precision and Recall

The alignment technique that we described in Section 3.4 uses an external alignment to decide how to interpret foreign words. Since this external alignment can be wrong, it is not completely trusted, using a parameter ξ' to allow for exploration.

A reasonable question is how to choose a good value for ξ'. It is easy to see that the values that give better results in terms of rate of successful interactions depend on the pragmatic precision of the external alignment α with respect to IM_1 and IM_2. If precision is high, agents should trust the alignment more, if it is low they should rely more on exploration.

To show this, we performed a small experiment, in which we analyzed the

rate of success of interactions between agents that use different values of ξ' and have alignments of different qualities. We used the customer and waiter agents from the example in Section 3.7.1 and let them interact for 150 times, measuring in how many cases they succeeded. As a simplification, we used only alignments that had the same values of precision and recall; this should be extended in future work to consider more varied values. We considered three alignment quality levels: low (precision and recall 0.2), medium (precision and recall 0.5) and high (precision and recall 0.8), and values of ξ' between 0.1 and 1.0. Figure 3.5 shows the results. As expected, when the alignment is good, best results are obtained with a high ξ', while for bad alignments it is better to make random choices. For medium quality, there is almost no difference, since the probability of a mapping being correct is similar to the one of choosing randomly the right option.

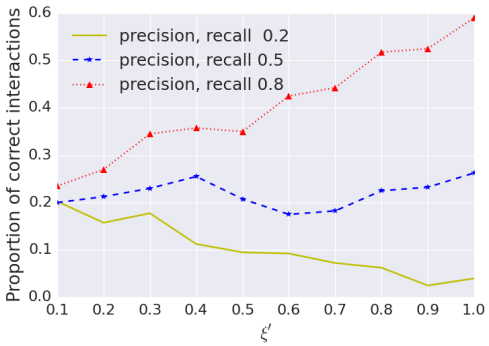


Figure 3.5: Success rates for different values of ξ'.

Estimating Pragmatic Precision and Recall

To compute pragmatic precision and recall two complete interaction are necessary. In distributed environments, it is not very likely to have this at hand. In what follows we discuss a different approach, in which agents use the experience of interaction to automatically estimate the values of precision and recall of an alignment. In this way they can evaluate alignments in a dynamic, distributed way, and at the same time improve their behavior using the ideas we just explained.

Let us first focus on estimating recall. In this case, agents can simply use the proportion of the mappings they made in successful interactions that were already in α.

$$recall_{est} = \frac{|\text{ mappings in successful interactions} \cap dom(\alpha)\ |}{|\text{ mappings in successful interactions }|}$$

Estimating precision is more complicated. A first attempt could be to consider:

$$precision_{est} = \frac{|\text{ mappings in successful interactions} \cap dom(\alpha)\ |}{|\text{ relevant mappings seen }|}$$

However, this considers as incorrect all the relevant mappings that were not part of successful interactions. This can under-estimate the precision, particularly in the first steps.

Alternatively, we propose to use a learning strategy that estimates the precision of α gradually, by analyzing which of the mappings that were made are likely to be correct and which ones are not. A possibility is to use the `evol` technique defined before.

To estimate precision, let *increased* be the set of all the mappings made that are in α and for which the calculated confidence is greater or equal to the one in α. Precision can then be estimated as:

$$precision_{est} = \frac{|\text{ increased} \cap dom(\alpha)\ |}{|\text{ relevant mappings seen }|}$$

This can improve the precision estimate in early stages, since mappings that are likely to be correct (because many good mappings were found after them) would still increase their value. These are preliminary ideas, that we plan to further develop and evaluate experimentally in future work.

3.8 Conclusion

We proposed methods that combine external vocabulary alignments, which can have been obtained from different sources, with interaction-based techniques. We show that they can significantly improve agents' vocabulary alignment, compared to using them separately. The most complex integration, in particular, shows how with simple techniques the detrimental effects of low quality of alignments can be mitigated. With respect to this, interesting conclusions about the quality of the alignments can be drawn from the experimentation. First, the level of recall seems to have more impact than the precision when the alignment is used for agent communication. This is worth exploring further, particularly given the current trend of favoring precision over recall in ontology alignment techniques (Dragisic et al. 2014).

The combination of experience-based methods and external alignments was simple after we formulated the method in the reinforcement learning paradigm. This also allowed us to identify a problem that made the alignment slow. Namely, not being able to learn from unsuccessful interactions, and it led us to propose a way of solving it. This new formulation is also useful to place the

technique in the spectrum of existing solutions, and to identify its relations with other approaches.

Additionally, we presented a way of evaluating external alignments in a particular interaction. We consider this to be a first step towards the development of ontology alignment tools that are particularly designed for agent interaction. These tools would require novel reasoning techniques that take into account contextual information about the tasks that are being performed to build mappings of high pragmatic precision and recall. To this aim, a first technical requirement is the formalization of a language that allows to express properties of the domain together with information about the interaction.

Chapter 4

Open Protocols

In Chapter 3 we considered protocols to be specified with finite state automata, as it is done in the work by Atencia and Schorlemmer (2012). This is a simple way of defining small, fixed interactions, however, it is too rigid for almost all applications. This is because it is necessary to explicitly specify everything that can be said at a given state, which can make the resulting protocols very rigid.

In this chapter we study how the ideas of interaction-based vocabulary alignment can be applied to a different type of protocols, that specify rules about what can be said instead of explicitly determining the flow of an interaction. Inferring alignments from this kind of protocols require different techniques, that we develop and evaluate here.

4.1 Introduction

Transition-based formalisms, such as finite state automata, are a simple technique to specify interaction protocols: at each state, some messages are allowed, and agents change states by sending messages. Moreover, they have a clear notion of satisfiability, which makes it easy to verify whether agents comply with them. However, although transition-based specifications may be useful for small interactions, they do not scale up, becoming very complicated when the interactions are large, or not very constrained. This is because finite state machines require to explicitly specify which messages can be sent at each point. If there are many possible messages, protocols become difficult to design, read and change. Moreover, these protocols describe a specific order in which messages should be sent, which can lead to over-specification. For these reasons the assumption that all agents share exactly the same structure of such a restrictive protocol, that is made in Chapter 3, is a strong one.

In this chapter we investigate interaction-based alignment techniques applied to a different type of protocols which provide more flexibility. These protocols,

that we call *open*, do not specify the messages that can be sent at each state, but instead define some general constraints that must be always satisfied. For example, instead of specifying that the waiter must ask *color?* after the customer asks for wine, these protocols could include a constraint saying that the color must be asked at some point in time after wine is ordered, leaving to the agent the decision of exactly when to do it. In this way, the protocol avoids determining when exactly this should happen, as well as everything else that can be said. Constraint-based protocols are usually specified with some kind of temporal logic, a useful way of talking about things that occur in time. For example, the protocols in (Giordano et al. 2007) use Dynamic Linear Temporal Logic.

In this dissertation we chose to use ConDec protocols (Pesic and van der Aalst 2006, Pesic et al. 2007), a simple formalism that uses linear temporal logic. Simply put, a ConDec protocol is a set of constraints. Since these constraints have much less information than a transition system, one protocol may not be enough to completely define the meaning of all words in a vocabulary. For this reason, in this chapter we study how agents can learn mappings when they perform different tasks. This idea is similar to the cross-situational language learning models proposed by Siskind (1996) and by Xu and Tenenbaum (2007). These works, that we discussed already in Chapter 2, investigate how word-meaning mappings can be inferred from observing different situations, each of them associated with a sentence. While our learning models are similar, we use interactions instead of visual images or pictures as situations. This implies also that agents learn an alignment that is useful across many different situations, and not for one specific protocol, as in Chapter 3.

The notion of complying with a constraint-based protocol is, in a way, inverse to the one for finite state machines: everything can be said, as long as it does not violate a constraint. For this reason, the techniques to learn an alignment from the experience of interacting are different for agents that use each of these specifications. In this chapter we develop probabilistic interaction-based alignment techniques. We consider agents that share the knowledge of how to perform a set of tasks but speak different languages, and we discuss how they can infer an alignment from interacting repeatedly. We propose a general technique that can be used for constraint protocols in general; and a more specific one, which uses the semantics of ConDec protocols in particular. We also show how agents can improve their learning by choosing intelligently the messages that they utter. We evaluate our techniques in a dataset of randomly generated protocols, and we compare them with a non-probabilistic method, that uses logical deduction. Finally, we consider a situation in which agents do not agree on exactly the same protocol structure, and we analyze if they can still find a useful alignment.

The remainder of this chapter is organized as follows. After introducing *open interaction protocols* in Section 4.2, in Section 4.3 we define a framework

for interacting with partners that use different vocabularies. Section 4.4 presents different techniques to learn an alignment from the experience of interacting as well as other ones that consider agents that actively want to learn. All methods can be used to learn alignments from scratch when there is no information, as well as to repair alignments obtained with other methods. We evaluate the different techniques experimentally in Section 4.5. Section 4.6 discusses agents that can use a particular class of *p-necessary* constraints in the learning process, and in Section 4.7 we compare our probabilistic learning technique to one based on logical inference. Finally, in 4.8 we discuss agents that do not share the structure of the protocols.

4.2 Open Interaction Protocols

As we mentioned, a way of gaining flexibility in protocol specifications is by constraining possible actions instead of defining a fixed procedure. *Constraint Declarative Protocols* (commonly known as ConDec protocols) are an example of these approaches. ConDec protocols were first proposed by Pesic and van der Aalst (2006) as a language to describe business protocols. In later work they have been used as a specification language for agent interactions, for example by Montali (2010), who presents an extension of the ConDec language and tools for its verification, and by Baldoni et al. (2010a) and Baldoni et al. (2013), who integrates ConDec constrains with commitment protocols, a framework to specify interactions with social semantics (Singh 2000). An important advantage of ConDec protocols is that they use linear temporal logic, a well-known formalization for which many reasoning tools are available. These protocols specify only the temporal aspect of the interaction, without any link to a more global view of the task that is being performed (see (Marengo et al. 2011) for a discussion on this), or to any other semantic information about the concepts that agents are talking about. In this chapter we show that temporal information is enough to infer an alignment between vocabularies that allows agents to communicate meaningfully. Of course, agents with richer protocols can incorporate the extra information to the learning process to make it faster or more precise.

The rest of this section is divided in two parts. In Section 4.2.1, we present the basic notions of ConDec protocols as they were defined in (Pesic and van der Aalst 2006). In Section 4.2.2 we present our adaptation to use ConDec protocols as specifications of interactions between agents that may use different vocabularies, as well as technical notions that we will need later.

4.2.1 Preliminaries

Linear temporal logic (LTL from now on) is a natural choice to express constraints about actions that occur in time. We will describe it following the

conventions used by Huth and Ryan (2004). The syntax of LTL consists of a finite set of propositional variables `Atoms`, the logical operators $\neg$, $\wedge$ and $\rightarrow$, and the temporal modal operators $\{\mathrm{G}, \mathrm{F}, \mathrm{X}, \mathrm{W}\}$.[1] The set of LTL formulas over `Atoms` is defined as follows:

1. if $p \in$ `Atoms`, then p is an LTL formula, and
2. if ϕ, ψ are LTL formulas, then $\neg\phi$, $\phi \wedge \psi$, $\phi \rightarrow \psi$, $\mathrm{G}\phi$, $\mathrm{F}\phi$, $\mathrm{X}\phi$, $\phi \,\mathrm{W}\, \psi$ are LTL formulas.

LTL formulas are interpreted over sequences of states, with each state being associated to a truth-valuation of the propositional variables in `Atoms`. The intuitive idea is that $\mathrm{G}\phi$ means that ϕ must be true in the truth-valuation of all following states, $\mathrm{F}\phi$ means that ϕ must be true eventually, $\mathrm{X}\phi$ means that ϕ must be true in the next state, and $\phi \, W \, \psi$ means that ϕ must be true until ψ is true, if that ever happens, or true always otherwise. Formally, consider a set of states S, and a labeling function $L : S \rightarrow 2^{\texttt{Atoms}}$ assigning each state to a subset of `Atoms`. For $s \in S$, the value of the propositional variables in $L(s)$ is interpreted as `true` in that state, and that of all variables in `Atoms` which are not in $L(s)$ is `false`. A word $w = s_1, s_2, \ldots$ is an infinite sequence of states in S, that is, an element in S^*. We write $w^{i:}$ for the suffix starting at s_i, that is, $w^{i:} = s_i, s_{i+1}, \ldots$

The semantics of LTL is defined as follows. Let $p \in$ `Atoms`, and let ϕ, ψ be two LTL formulas.

- $w \models p$ if $p \in L(s_1)$
- $w \models \neg\phi$ if $w \not\models \phi$
- $w \models \phi \wedge \psi$ if $w \models \phi$ and $w \models \psi$
- $w \models \phi \rightarrow \psi$ if $w \models \psi$ whenever $w \models \phi$
- $w \models \mathrm{G}\phi$ if for all $i \geq 1$, $w^{i:} \models \phi$
- $w \models \mathrm{F}\phi$ if there is some $i \geq 1$ such that $w^{i:} \models \phi$
- $w \models \mathrm{X}\phi$ if $w^{2:} \models \phi$
- $w \models \phi \,\mathrm{W}\, \psi$ if either there is some $i \geq 1$ such $w^{i:} \models \psi$ and for all j such that $1 \leq j \leq i - 1$ we have $w^{j:} \models \phi$, or for all $k \geq 1$ we have $w^{k:} \models \phi$

[1] This set is not minimal, but it is the most useful one for our purposes. For the same reason we use weak until (W) instead of until (U).

A set of LTL formulas, called a *theory*, is *satisfiable* if there exists a sequence for which all the formulas are true. In that case, the sequence is a *model* of the theory. The satisfiability problem in LTL is decidable, as well as the *model checking* problem, which consists in deciding if a given sequence is a model of a theory.

The ConDec specification language provides a set of constraint templates, which rename some particular basic LTL formulas. The first column of Table 4.1 shows the ones that we use in this chapter. We use a simplified version[2] of the original templates that were introduced by Pesic and van der Aalst (2006) and by van der Aalst and Pesic (2007). The most similar version is the one proposed by Baldoni et al. (2010b). A ConDec protocol, then, is a set of constraints over a particular set of variables.

Definition 4.1. (ConDec protocol) Let M be a set of propositional variables. We call $Cons(M)$ the set of all ConDec constraints over M, that is, the templates in the first column of Table 4.1 for any $m, m' \in M$ and $n \in \mathbb{N}^+$. A *ConDec protocol over* M is a finite subset of $Cons(M)$. ■

The LTL meanings of the constraints can be seen in the second column of Table 4.1. In general, constraints are classified into two types. Existential constraints (*existence* and !*existence*) predicate over how many times some action can be performed. Relational constraints (the remaining ones) describe binary relations between two actions. Constraints can be positive or negative, representing obligations or prohibitions respectively. Negative constraints start with an exclamation mark.

4.2.2 Open Protocols as Interaction Protocols

We can now define *interaction protocols*, which constrain the way in which agents can utter messages. Intuitively, an interaction protocol is a ConDec protocol where variables are utterances.

Definition 4.2. (Interaction protocol) Let a *vocabulary* V be a set of words, and A be a set of agent IDs. $M = A \times V$ is the set of *messages* over V and A. An *interaction protocol* is a ConDec protocol $\mathfrak{P} \subseteq Cons(M)$. ■

The semantics of an interaction protocol is defined over interactions that represent a sequence of uttered messages.

Definition 4.3. Let M be a set of messages. An *interaction* over M is a finite sequence $I \in M^*$. ■

[2] Since we are not interested in the usability of the protocols here, we do not include those constraints that work as syntactic sugar, such as $exactly(n, a)$, that can be replaced by including $existence(n, a)$ and $!existence(n, a)$.

Constraint	LTL meaning
$existence(1, m)$	$\mathrm{F}m$
$existence(n+1, m)$	$\mathrm{F}(m \wedge \mathrm{X}existence(n, m))$
$!existence(n, m)$	$\neg existence(n, m)$
$correlation(m, m')$	$\mathrm{F}m \rightarrow \mathrm{F}m'$
$!correlation(m, m')$	$\mathrm{F}m \rightarrow \neg\mathrm{F}m'$
$response(m, m')$	$\mathrm{G}(m \rightarrow \mathrm{F}m')$
$!response(m, m')$	$\mathrm{G}(m \rightarrow \neg\mathrm{F}m')$
$before(m, m')$	$\neg m' \,\mathrm{W}\, m$
$!before(m, m')$	$\mathrm{G}(\mathrm{F}m' \rightarrow \neg m)$
$premise(m, m')$	$\mathrm{G}(\mathrm{X}m' \rightarrow m)$
$!premise(m, m')$	$\mathrm{G}(\mathrm{X}m' \rightarrow \neg m)$
$imm\_after(m, m')$	$\mathrm{G}(m \rightarrow \mathrm{X}m')$
$!imm\_after(m, m')$	$\mathrm{G}(m \rightarrow \mathrm{X}\neg m')$

Table 4.1: LTL definitions of constraints, where $m, m' \in M$ and $n \in \mathbb{N}^+$.

Any operation over sequences can be applied to an interaction I. We will use the length function ($len(I)$), the function that appends an element ($I \,.\, m$), and the notion of prefix (noted $I \preccurlyeq I'$ if I is a prefix of I'). As in the previous chapter, $I(i)$ will denote the element in the i-th position in I. From now on, let V and A be, respectively, a vocabulary and a set of agent ids, and $M = A \times V$ be a set of messages including a message m. Let also $\mathfrak{P}$ and I be, respectively, a protocol and an interaction over M.

Since interaction protocols are essentially LTL theories, we will define their semantics using the existing definitions for that logic. To this end it is only necessary to encode interactions into infinite sequences of states that are labeled with subsets of M. Let S be a set of states and the labeling function $L : S \rightarrow 2^M$.

Definition 4.4. (Satisfiability) Let w_I be the infinite sequence of states in S such that:

1. $L(w_I(i)) = \{I(i)\}$ for $1 \leq i \leq len(I)$
2. $L(w_I(i)) = \varnothing$ for $i > len(I)$

We call I a *model* of $\mathfrak{P}$ (noted $I \models \mathfrak{P}$) if $w_I \models \mathfrak{P}$. We call I a *partial model* of $\mathfrak{P}$ (noted $I \models_p \mathfrak{P}$) if it is a prefix of a model of $\mathfrak{P}$, that is, there exists I'

such that $I \preccurlyeq I'$ (I is a prefix of I') and $I' \models \mathfrak{P}$. Sometimes we will refer to the satisfiability of a single constraint c. In those cases we will use $I \models c$ for $I \models \{c\}$ and $I \models_p c$ for $I \models_p \{c\}$. ■

Definition 4.4 implies that checking satisfiability of an interaction protocol is equivalent to checking LTL satisfiability, and is therefore decidable.

As already mentioned, we are interested in agents that use different vocabularies, but share the *knowledge of how to perform a task*. In the rest of this section we define more precisely what this means. Let us start by defining the notion of vocabulary translation. From now on, let V' be another vocabulary, and $M' = A \times V'$ a set of messages with the same agents than M, but vocabulary V'. Let also $\mathfrak{P}' \subseteq Cons(M')$ be a protocol.

Definition 4.5. A *translation* between V and V' is a function $\tau : V \to V'$.

τ can be extended homomorphically to a function between:

- messages in $M = A \times V$ and in $M' = A \times V'$ ($\tau : M \to M'$)
- constraints over M and M' ($\tau : Cons(M) \to Cons(M')$)
- interactions over M and M' ($\tau : M^* \to M'^*$) and sets of interactions ($\tau : 2^{M^*} \to 2^{M'^*}$)
- protocols over M and M' ($\tau : 2^{Cons(M)} \to 2^{Cons(M')}$) ■

We will call *mapping* a pair of words form different vocabularies. We will say a mapping between words $v \in V$ and $v' \in V'$ is *correct* for a translation τ if $\tau(v) = v'$, or incorrect otherwise. Note that, unlike previously, the translation is not anymore parametrized by a state. This is a different conceptualization of translations. While this approach cannot express the contextualization of mappings as before, the obtained translation is more general.

As in Chapter 3, we capture the idea of *sharing the knowledge of how to perform a task*, with the notion of *compatibility* between protocols. In this case, it consists simply in having the same models modulo a translation.

Definition 4.6. Let $Int(\mathfrak{P}) = \{I \in M^* \text{ such that } I \models \mathfrak{P}\}$ be the set of models of a protocol $\mathfrak{P}$. The protocols $\mathfrak{P}$ and $\mathfrak{P}'$ are *compatible* if there exist translations $\tau : V \to V'$ and $\tau' : V' \to V$ such that

$$Int(\mathfrak{P}) = \tau(Int(\mathfrak{P}'))$$

$$Int(\mathfrak{P}') = \tau'(Int(\mathfrak{P}))$$

In many cases we will be working with a bijective translation τ such that $\tau^{-1} = \tau'$. If we know the translations τ and τ' for which this condition holds, we can say they are *compatible under* τ. ■

Example (*ordering drinks*). Let us formalize ConDec protocols for an interaction to order drinks. Consider again a waiter W and a customer C interacting to order drinks. Let the vocabulary of the customer be $V_C = \{\mathsf{to\ drink}, \mathsf{beer}, \mathsf{wine}, \mathsf{water}, \mathsf{size}, \mathsf{pint}, \mathsf{half\ pint}\}$, and that of the *Waiter* be $V_W = \{\mathsf{da\ bere}, \mathsf{birra}, \mathsf{vino}, \mathsf{acqua}, \mathsf{tipo}, \mathsf{media}, \mathsf{piccola}\}$. Consider the bijective translation $\tau : V_W \to V_C$ such that $\tau(\mathsf{da\ bere}) = \mathsf{to\ drink}$, $\tau(\mathsf{birra}) = \mathsf{beer}$, $\tau(\mathsf{vino}) = \mathsf{wine}$, $\tau(\mathsf{acqua}) = \mathsf{water}$, $\tau(\mathsf{tipo}) = \mathsf{size}$, $\tau(\mathsf{media}) = \mathsf{pint}$, $\tau(\mathsf{piccola}) = \mathsf{half\ pint}$.

The following protocols can specify the *ordering drinks* interaction. $\mathfrak{P}_C$ is the protocol used by the customer, with English messages, while $\mathfrak{P}_W$ is the one of the waiter, in Italian. Both agents agree on the first six constraints, which describe what can happen after the waiter asks what the customer wants to drink. For example, the fourth constraint states that if the customer orders beer, then at some point after that the waiter has to ask for the size (or *tipo* in Italian, in this particular interaction). The protocol of the waiter, however, has one extra constraint, stating that beer and wine cannot be ordered together. This is not a constraint for the customer.

$$\begin{gathered}\mathfrak{P}_C = \{\mathit{existence}(1, \langle W, \mathsf{to\ drink}\rangle),\\ \mathit{premise}(\langle W, \mathsf{to\ drink}\rangle, \langle C, \mathsf{beer}\rangle),\\ \mathit{premise}(\langle W, \mathsf{to\ drink}\rangle, \langle C, \mathsf{wine}\rangle),\\ \mathit{response}(\langle C, \mathsf{beer}\rangle, \langle W, \mathsf{size}\rangle),\\ \mathit{premise}(\langle W, \mathsf{size}\rangle, \langle C, \mathsf{half\ pint}\rangle),\\ \mathit{premise}(\langle W, \mathsf{size}\rangle, \langle C, \mathsf{pint}\rangle)\}\end{gathered}$$

$$\begin{gathered}\mathfrak{P}_W = \{\mathit{existence}(1, \langle W, \mathsf{da\ bere}\rangle),\\ \mathit{premise}(\langle W, \mathsf{da\ bere}\rangle, \langle C, \mathsf{birra}\rangle),\\ \mathit{premise}(\langle W, \mathsf{da\ bere}\rangle, \langle C, \mathsf{vino}\rangle),\\ \mathit{response}(\langle C, \mathsf{birra}\rangle, \langle W, \mathsf{tipo}\rangle),\\ \mathit{premise}(\langle W, \mathsf{tipo}\rangle, \langle C, \mathsf{piccola}\rangle),\\ \mathit{premise}(\langle W, \mathsf{tipo}\rangle, \langle C, \mathsf{media}\rangle),\\ !\mathit{correlation}(\langle C, \mathsf{birra}\rangle, \langle W, \mathsf{vino}\rangle)\}\end{gathered}$$

The two protocols above are not compatible under any translation. The protocol $\mathfrak{P}_C$ has a model in which the customer orders wine *and* beer, while $\mathfrak{P}_W$, due to the last constraint, only accepts as models interactions in which only one alcoholic beverage is ordered. If $!\mathit{correlation}(\mathsf{beer}, \mathsf{wine})$ is added to $\mathfrak{P}_C$, the resulting protocols would be compatible, in particular under τ.

The notion of compatibility will be a key one in our techniques. If agents have compatible protocols, at each state they agree on which messages are possible modulo a translation, provided they interpreted correctly the previous messages. This can be used to obtain information about possible mappings. Concretely, when an agent receives a message it will analyze each possible interpretation to see whether choosing it results in an interaction that is consistent with the

constraints, extracting information about the mappings from this. The following defines formally the notion of not being consistent with a constraint.

Definition 4.7. (Constraint violation) Let $c \in Cons(M)$ be a constraint. We say c is *violated* by I if $I \not\models_p c$. ■

It is important to note that we say that a constraint is violated when the interaction does not *partially* satisfy it. This is because we are interested in the situation for which a message is definitely not possible. For example, we may have $I \,.\, \langle a_2, v_1\rangle \not\models response(\langle a_2, v_1\rangle, \langle a_1, w_1\rangle)$, but this does not mean that a_2 cannot say v_1, because a_1 could later say w_1 making the interaction satisfy the constraint. A constraint is only violated when no subsequent utterance can make the interaction satisfy it. For some constraints (such as *response*) this is never possible. These constraints can only be identified as not satisfied when observing complete interactions. Other constraints can be violated by a message before the interaction has finished. The concept of *p-necessity* captures this difference.

Definition 4.8. Let $c \in Cons(M)$ be a constraint. We say c is *p-necessary* when, for any interaction I over M, $I \models_p c$. Inversely, c is *non-p-necessary* if there exists an interaction I such that $I \not\models_p c$. ■

We will call *(non)-p-necessities* to (non)-p-necessary constraints. In words, a constraint is p-necessary when, even if it is not satisfied by a given interaction, it is partially satisfied by it, i.e, it can always eventually be satisfied by a continuation of the interaction. A constraint is non-p-necessary if there exists an interaction that cannot be continued in any way that satisfies it. In fact, as we will show, this happens for every interaction that is not a model of the constraint. That is, for non-p-necessities, all interactions that are not models are also not partial models.

The following proposition divides the constraints in our protocols between those that are p-necessary and those that are not.

Proposition 4.1. *The constraints existence, correlation, and response are p-necessary. All other constraints are non-p-necessary.*

Proof. Let $\#(m, I)$ be the number of times that message m appears in I. We will use $I^{i:}$ to denote the suffix of I starting at index i, and $I^{:i}$ for the prefix that finishes at index i. Let us first prove p-necessity. We need to show that $I \models_p c$ for any interaction I. If $I \models c$, then $I \models_p c$ trivially. We consider, for each p-necessary constraint c, an interaction I such that $I \not\models c$, and show that $I \models_p c$.

- For $c = existence(n, m)$, if $I \not\models c$, then $\#(m, I) < n$. Let I' be an interaction such that $\#(m, I') = n - \#(m, I)$, and consider a new interaction I''

composed of I followed by I'. Then $I \prec I''$, and $\#(m, I'') = n$, so $I'' \models c$, and therefore $I \models_p c$.

- For $c = \mathit{correlation}(m, m')$, if $I \not\models c$, then either $m \in I$ and $m' \notin I$ or vice versa. Suppose $m \in I$; the other case is analogous. Let $I' = I \,.\, m'$. Then $I \prec I'$, and both $m \in I'$ and $m' \in I'$, so $I' \models c$, and therefore $I \models_p c$.

- For $c = \mathit{response}(m, m')$, if $I \not\models c$, then there exists i, $1 \leq i \leq \mathit{len}(I)$, such that $I(i) = m$ but $m' \notin I^{:i}$. Taking $I' = I \,.\, m'$ as in the previous case also works as proof here.

To prove non-p-necessity, we need to show an example of an interaction I such that, for any I', if $I \prec I'$ then $I' \not\models c$. As we show, it is enough to consider any I such that $I \not\models c$:

- If $c = \mathit{before}(m, m')$ and $I \not\models c$, there is an index i, $1 \leq i \leq \mathit{len}(I)$, such that $I(i) = m'$, but $m \notin I^{:i}$. Then this is also true for any I' if $I \prec I'$.

- If $c = \mathit{premise}(m, m')$ and $I \not\models c$, then either $I(1) = m'$ or there is an index i, $2 \leq i \leq \mathit{len}(I))$ such that $I(i) = m'$ and $I(i-1) \neq m$. In any case, this holds for any I' such that $I \prec I'$.

- If $c = !\mathit{response}(m, m')$ or $c = !\mathit{before}(m, m')$ and $I \not\models c$, there are indexes i, j $(1 \leq i < j \leq \mathit{len}(I))$ such that $I_i = m$ and $I_j = m'$. This is also true for any I' if $I \prec I'$.

- If $c = !\mathit{correlation}(m, m')$ and $I \not\models c$, then $m \in I$ and $m' \in I$. If $I \prec I'$, then also $m \in I'$ and $m' \in I'$.

- If $c = !\mathit{premise}(m, m')$ or $c = !\mathit{imm\_after}(m, m')$ and $I \not\models c$, there is $I(j) = m$ and $I(j+1) = m'$ for some $1 \leq j < \mathit{len}(I)$. If $I \prec I'$, this is also true in I'.

- If $c = !\mathit{existence}(n, m)$ and $I \not\models c$, then $\#(m, I) > n$. This is the same for any I' if $I \prec I'$.

□

Corollary 4.1. *Let c be a non-p-necessary constraint and I an interaction. If $I \not\models c$, then $I \not\models_p c$.*

Proof. It is derived from the proof of the proposition, since we did not impose any restriction on the interaction I chosen as an example. □

4.3 Communicating with Heterogeneous Partners

The communicative that we study in this chapter is similar to the one in Chapter 3, but now we consider agents that perform different tasks together. From now on, we consider interactions between two agents a_1 and a_2, who need to collaborate with each other to perform certain tasks. They both know the tasks to perform, but use different vocabularies V_1 and V_2 respectively. The objective of the methods we propose is to allow agents to learn to understand each other, enabling them to perform the tasks together. Again, we consider only interactions between two agents, each of whom speaks only one vocabulary, but this does not imply that all interactions need to be with the same partner. As long as the foreign vocabulary remains the same, an agent can change its interlocutor and keep learning. If there is more than one foreign vocabulary, agents need to know which one is used by each interlocutor to apply our method directly.

The translation technique that we propose is based on the assumption that, for each task, both agents share the procedural knowledge that describes how it must be performed. This can be made more concrete as follows:

1. Both agents agree on a set of tasks that they can perform together.
2. For each task, agents have protocols that are compatible with each other.
3. Agents use their vocabulary consistently throughout different tasks, i.e., the protocols for all tasks are compatible under the same translation.

Formally, let $M_1 = \{a_1, a_2\} \times V_1$ and $M_2 = \{a_1, a_2\} \times V_2$ be sets of messages. Let the function $\tau : V_1 \to V_2$ be a bijective translation such that, for each task that a_1 and a_2 can perform together, they have protocols $\mathfrak{P}_1$ and $\mathfrak{P}_2$ over M_1 and M_2 respectively, and $\mathfrak{P}_1$ and $\mathfrak{P}_2$ are compatible under τ. We assume that whenever agents interact, they use protocols that correspond to the same task, or more concretely, compatible protocols.

Note that we assume that the translation τ is bijective. Recall that we defined compatibility as equivalence of all models modulo a translation. If this translation is not bijective, and there are, for example, two words v_1 and v_1' in V_1 mapped to v_2 in V_2, these two words would have to be indistinguishable at least in the context of the protocols that are considered. If there is a situation in which, for example, v_1 can be said and v_1' cannot, the protocols could not be compatible under τ. Therefore, in practical terms, in that context it would be equivalent to consider a vocabulary V_1' with only one of the words. For this reason, and since it simplifies the exposition significantly, we simply assume that the translation is bijective. This assumption, however, is not necessary for our techniques to work as we report. We will discuss at each point how to handle non-bijective translations.

During an interaction, an agent can send messages using words of its vocabulary and receive others from its interlocutor, or finish the communication if certain conditions hold. We do not force them to follow any particular turn-taking pattern, but we assume that they both know who speaks at each point. We also assume messages are not lost and arrive on time, at least most of the times. Since we propose a probabilistic approach, we can handle a certain amount of errors, but we do not analyze this aspect in depth here.

Like in the previous chapter, the receiver needs to decode the words it gets, interpreting them in its own terms to be able to continue the execution. To this end, agents need to learn the translation that allows them to interpret received words correctly, that is, the τ under which their protocols are compatible. We present a general approach to learn τ from the experience of interacting to perform different tasks sequentially. For simplicity, from now on we adopt the perspective of agent a_1. Since the objective of this agent is to translate words from V_2, it needs to find the function $\tau : V_2 \to V_1$.

We propose to learn τ by taking into account which messages are allowed by the protocols at each state. When a_1 receives a message in the vocabulary V_2 used by a_2, it analyzes which interpretations are allowed by the protocol and which ones are not, and learns from this information. To this end, we use a probabilistic learning technique, in which agents maintain a confidence distribution over possible meanings for foreign words, and update their values with the experience of interacting. Probabilistic techniques are commonly used for all kind of learning problems and particularly for learning word-meaning correspondences (see for example (Xu and Tenenbaum 2007)), and, as we will explain, they are useful to capture the uncertainty that agents encounter during the learning process. In our problem, an alternative to using probabilistic techniques is possible. It consists on storing relevant information about every observed interaction, and then extracting logical consequences from it. In Section 4.7 we compare our techniques with one that only uses logical inference. Our probabilistic technique will use the notion of *alignment*, defined in Chapter 3 and repeated here.

Definition 4.9. Let V_1 and V_2 be two vocabularies. An *alignment* is a partial function $\omega : V_1 \times V_2 \nrightarrow [0, 1]$. ■

Intuitively, if $v_1 \in V_1$ and $v_2 \in V_2$, the value $\omega(v_1, v_2)$ represents a_1's confidence in that v_1 is the correct meaning for v_2, that is, that $\tau(v_2) = v_1$. The function is partial because a_1 does not know V_2 a priori, and therefore it is only defined for those words that the agent received in a message at some point. For the same reason we will keep $\sum_{v \in V_1} \omega(v, v_2) = 1$ but do not require the same in the other direction.

In the following section we will explain in detail how ω is updated with the agent's interacting experiences. Now, let us focus on the dynamics of the

interaction; concretely, on how agents send messages, interpret received ones, and finish the interaction.

As we discussed, agents send messages in their own vocabularies that the receivers have to interpret. This implies that the sequence of messages that were actually sent differs from the interaction that each agent interpreted. We will call the former ones *global interactions*, where messages can either be $\langle a_1, v_1 \rangle$ or $\langle a_2, v_2 \rangle$ with $v_1 \in V_1$ and $v_2 \in V_2$. *Local interactions* are the ones interpreted by agents and contain only messages with words in V_1 or only messages with words in V_2. We will call I_1 and I_2 the local interactions of a_1 and a_2 respectively. To explain the actions of choosing utterances, interpretations, and when to finish the interaction we will make use of the notion of *possible messages*. That is, those messages that can be uttered at a certain point in the interaction.

Definition 4.10. Let M be a set of messages and $\mathfrak{P}$ a protocol over it. Let I be a local interaction that has happened so far (and $I \models_p \mathfrak{P}$). We call the *possible messages* at that point all $m \in M$ such that when they are added to I, the interaction remains a partial model, that is $I \,.\, m \models_p \mathfrak{P}$. ■

Notice, first, that this definition of possible messages is different to the one used in Chapter 3. Here, any message is possible as long as it does not violate a constraint. Second, note that we use the notion of partial satisfiability to define possible messages. That is, a message is possible if, after being uttered, the interaction can be finished in a way that satisfies the protocol. Finally, possible messages for a_1 are not necessarily the translation of the ones for a_2. This is because the local interactions of each agent depend on their interpretations of past messages, which are not necessarily correct.

Now we can explain the dynamics of the interaction.

Choosing messages to send The choice of messages to utter is internal to each agent and depends on its goals while interacting. We do not impose any restriction on this, except for not violating the constraints in the protocol. That is, agent a with protocol $\mathfrak{P}$ can utter a word v after interaction I only if $\langle a, v \rangle$ is a possible message for I and $\mathfrak{P}$.

Choosing interpretations When agent a_1 receives a message $v_2 \in V_2$ from a_2, it needs to interpret it in V_1 to be able to continue the interaction. To this aim, agents use the information in the alignment ω. Based on τ's bijectivity we assume that, in an interaction, agents choose always the same mapping for a foreign word, and they do not choose the same mapping for two different words. To formalize this we define the function μ, that represents the mappings made in an interaction.

Definition 4.11. Let I be a global interaction and I_1 agent a_1's corresponding local interaction. The partial function $\mu : V_2 \nrightarrow V_1$ maps all the foreign words received by a_1 in an interaction to the local ones chosen as interpretations. That is, $\mu(v_2) = v_1$ if and only if there exists an index i such that $1 \leq i \leq len(i)$, $I(i) = \langle a_2, v_2 \rangle$, and $I_1(i) = \langle a_2, v_1 \rangle$. ■

Note that if the interaction has not started ($I = I_1 = []$), then $dom(\mu) = \varnothing$. If agent a_1 receives v_2 after local interaction I_1, it will choose a word $v \in V_1$ only if forms a possible message when uttered by a_2, and if it was not already chosen as an interpretation before. Between the words that satisfy those conditions, it chooses (randomly) one that has highest confidence. Let W be all $v \in V_1$ such that $\langle a_2, v \rangle$ is possible, and such that $v \notin img(\mu)$. Let *random* be a function that selects a random element from a set. The agent will choose an interpretation for v_2 as follows:

$$\begin{array}{ll} random(\underset{v \in W}{\operatorname{argmax}}\, \omega(v, v_2)) & \text{if } v_2 \notin \mathrm{dom}(\mu) \\ \mu(v_2) & \text{if } v_2 \in \mathrm{dom}(\mu) \text{ and } \langle a_2, \mu(v_2) \rangle \text{ is possible} \end{array}$$

In the remaining case, when $v_2 \in \mathrm{dom}(\mu)$ but $\langle a_2, \mu(v_2) \rangle$ is not possible, the agent considers the interpretation failed and it finishes the interaction. Note that the way in which agents choose interpretations makes μ injective. In the case when τ is non-bijective, agents should choose an interpretation without taking μ into account.

Finishing the interaction An interaction can finish in two ways. First, an interaction always finishes when an agent has no possible messages to say. An agent can also finish the conversation whenever it wants if it considers that the interaction is successful, i.e., if it is a model (not partial) of the protocol when it ends. In this case, it simply stops talking, producing a time-out that lets the other agent realize the interaction has finished.

4.4 Learning Translations from Interactions

As we mentioned earlier, the alignment ω is updated with the experience that agents collect when interacting. More concretely, agents decide how to update the alignment by analyzing whether an interpretation is possible or not for a received word. Recalling Definition 4.8, the constraints that can be violated by a new message are those that are not p-necessary. In this chapter we will focus mainly on the information obtained from this kind of constraints. In Section 4.6 we discuss how we could also take into account p-necessary constraints, and we discuss why this is a more challenging problem.

The approach that we propose to update ω can incorporate external alignments, as it is done in Chapter 3 for the case of protocols specified with finite state machines. As before, these alignments represent previous knowledge about the foreign vocabulary, which can have been obtained from previous experience, from a matching tool (Euzenat and Shvaiko 2013), or with techniques based on semantic vectors (see for example the work by Pennington et al. (2014)). As we already discussed, external alignments can never be completely trusted, either becuse of technical reasons or because they compute mappings that are not adequate for the context of the task being performed. For example, an Italian-English dictionary might translate piccola as small and media as half (or medium). Therefore, the mapping between half pint and piccola would hardly if ever be found. In an ordering drinks situation, however, this is the mapping that makes the customer obtain the size of beer she wants.

When external alignments are available, the techniques in this chapter can be applied as methods for testing and repairing possibly wrong mappings between to words. In general, existing tools provide a function that assigns confidence values to mappings, just like our alignments (see Definition 4.9). An external alignment for agent a_1 is, then, an alignment $\alpha : V_1 \times V_2' \nrightarrow [0,1]$, where V_2' is a set of terms such that $V_2' \cap V_2 \neq \varnothing$. The alignment α does not need to be defined over V_2, but even if it is not, the information it provides about words in that set can still be used. We interpret the confidences always positively when the confidence is greater than 0, meaning that a mapping with low confidence has always more confidence than one that does not exist. If the agent has an alignment that maps words between them instead of assigning values (a translation, as in Definition 4.5), it can be converted into an alignment by assigning value 1 to each correct mapping.

External alignments are taken into account in the initialization of the values of ω, which happens the first time an agent receives each foreign word. When a_1 receives message $\langle a_2, v_2 \rangle$ for the first time, it assigns a value $\omega(v, v_2)$ for each $v \in V_1$. This is done in a way such that it takes into account the previous alignment, and no word has value 0 or 1, considering in this way that the external mappings could be incorrect. If a_1 has an external alignment α and $v_2 \in dom(\alpha)$, it updates ω using that information. For each $v_1 \in V_1$,

$$\omega(v_1, v_2) = \begin{cases} \alpha(v_1, v_2) & \text{if } (v_1, v_2) \in dom(\alpha) \\ 0 & \text{otherwise} \end{cases}$$

Then it normalizes the values using an exponential method (we use *softmax*). This ensures that the values for all mappings start in the open interval $(0, 1)$. If there is no external alignment, the agent initializes the interpretation values with a uniform distribution, setting $\omega(v_1, v_2) = \dfrac{1}{\mid V_1 \mid}$ for all $v_1 \in V_1$.

After the initialization phase, or whenever a_1 receives a word v_2 that it has already received before, the update phase begins. For each word $v_1 \in V$, the value $\omega(v_1, v_2)$ will be updated according to whether v_1 is or not a possible interpretation. The technique that we propose to this aim is inspired in Bayesian updates, combining the previous value of the mapping in ω and the information about which interpretations are possible in the current interaction. Intuitively, if the word v_2 is received after interaction I_1, we want $\omega(v_1, v_2)$ to express the probability that v_1 is the correct interpretation for v_2 in this situation, that we note $p(\tau(v_2) = v_1 \mid I_1 \,.\, \langle a_2, v_2 \rangle)$. According to Bayes' rule this can be expressed as follows:[3]

$$p(\tau(v_2) = v_1 \mid I_1 \,.\, \langle a_2, v_2 \rangle) = p(I_1 \,.\, \langle a_2, v_2 \rangle \mid \tau(v_2) = v_1)\; p(\tau(v_2) = v_1) \tag{4.1}$$

Let us analyze this expression. First, $p(\tau(v_2) = v_1)$ is the prior probability that v_1 is a correct interpretation of v_2. This information can be obtained from the alignment $\omega(v_1, v_2)$. Since we keep the function normalized for $v \in V_1$, we can use the value directly. Second, the value $p(I_1 \,.\, \langle a_2, v_2 \rangle \mid \tau(v_2) = v_1)$ represents the probability that v_2 is said by a_2 after I_1 if the correct translation of v_2 is v_1. This is equivalent to the probability that v_1 is said after I_1. Therefore we can rewrite Equation 4.1 as:

$$p(\tau(v_2) = v_1 \mid I_1 \,.\, \langle a_2, v_2 \rangle) = p(I_1 \,.\, \langle a_2, v_1 \rangle)\; \omega(v_1, v_2) \tag{4.2}$$

Since it only involves words in the local vocabulary, the probability that a_2 says v_1 after I_1 can be computed as follows:

$$p(I_1 \,.\, \langle a_2, v_1 \rangle)) = \begin{cases} 1 & \text{if } \langle a_2, v_1 \rangle \text{ is possible for } I_1 \\ 0 & \text{otherwise} \end{cases} \tag{4.3}$$

These updates are only meaningful if agents are completely sure that they interpreted every foreign message up to I_1 correctly and, moreover, that the same is true for their interlocutor. That is, it is necessary that $\tau(I_2) = I_1$. Taking this not into account can be problematic when a message $\langle a_2, v_1 \rangle$ is not possible. According to Equation 4.3, the probability that v_2 should be interpreted as v_1 is 0. However, it might be that $\tau(v_2) = v_1$, but that the message $\langle a_2, v_1 \rangle$ is not possible because of a previously misinterpreted word, either by a_1 or a_2. This happens, for example, if $\mathfrak{P}_1$ has a constraint $!response(\langle a_2, v_1' \rangle, \langle a_2, v_1 \rangle)$ and a_1 interpreted a word incorrectly as v_1' before. Since it is not possible to ensure that previous messages were correctly interpreted by everyone, we present a way of taking this uncertainty into account when updating ω. To this aim we introduce

[3] If τ is bijective, the denominator in Bayes' rule, also known as the *marginal probability* of $I_1 \,.\, \langle a_2, v_2 \rangle$, can be omitted directly, since $\sum_{v \in v_1} p(I_1 \,.\, \langle a_2, v_2 \rangle \mid \tau(v_2) = v) p(\tau(v_2) = v) = 1$. Otherwise, it should be included as a normalizing constant.

the distribution p_{sh}, that represents the confidence that the interaction is shared, that is, that $\tau(I_2) = I_1$. If the interaction is not shared, no inference can be made from the observation, and the probability should be kept as $\omega(v_1, v_2)$. Now the alignment is updated by considering both possibilities, weighted by p_{sh}:

$$p(\tau(v_2) = v_1 \mid I_1.\langle a_2, v_2\rangle) = p(I_1.\langle a_2, v_1\rangle)\ \omega(v_1, v_2)\ p_{sh}(I_1) + \omega(v_1, v_2)\ (1 - p_{sh}(I_1))$$

As we already mentioned, p_{sh} is impossible to compute exactly. This is because agents have no access to the alignment of their partners, and therefore cannot assign a probability to the words that they interpreted. In Section 4.4.1 we show two different ways of choosing p_{sh}. Another particularity of our learning setup is that agents contribute directly into building part of the evidence, since they send the messages that will later make up the interaction. In Section 4.4.2 we discuss how agent can exploit this to learn faster. The performance of these agents is evaluated in the next section.

4.4.1 Estimating the Confidence in the Interaction

We now present different ways of estimating p_{sh}, the confidence that the interaction is shared, to take it into account in the updates. The first method is independent of the specific constraints in ConDec protocols, and can be used for any constraint system for which satisfiability can be computed. The second one takes into account the particularities of each constraint to make a more fine-grained update.

Simple Strategy

A first straightforward approach consists in setting the value of $p_{sh}(I)$ to a constant for any I. This method only takes into account the already defined notion of *possible messages*, without any reasoning on the semantics of ConDec constraints. Therefore, it can be applied to any kind of logical specification, provided that there is a method to decide if an interaction is a model of a protocol. Formally, let $r \in [0, 1]$ be a punishment parameter. The update for each $v \in V_1$, when v_2 is received after interaction I, is as follows:

$$\omega(v_1, v_2) := \begin{cases} \omega(v_1, v_2)\ (1 - r) & \text{if } \langle a_2, v_1\rangle \text{ is not possible} \\ \omega(v_1, v_2) & \text{otherwise} \end{cases} \qquad \text{(Simple Agent)}$$

After all the updates for a given message are completed, the values are normalized in such a way that $\sum_{v \in V_1} \omega(v, v_2) = 1$.

Reasoning Strategy

The other method we propose is a way of considering the semantics of ConDec constraints in particular, and to use the extra information they contain. To illustrate the importance of this information, recall the *ordering drinks* example, and particularly the waiter's protocol $\mathfrak{P}_W$. Consider an interaction in which the customer said water but the waiter interpreted it as vino (meaning *wine* in Italian). If the customer says beer, the waiter may think that interpreting it as birra (meaning *beer*) is impossible, since ordering two alcoholic beverages is not allowed (!*correlation*(birra, vino) would be violated). However, this is only due to having misunderstood water in the first place. We now present an approach that lets agents make use of the semantics of violated constraints to perform a more fine-grained update of the alignment ω.

If a_1 finds that v_1 is not a possible interpretation for v_2, consider all constraints $c \in \mathfrak{P}_1$ that are violated by $\langle a_2, v_1 \rangle$. The idea is to identify, for each violated constraint, which are the mappings that can be wrong, thus leading to the violation. The mapping between v_1 and v_2 is always a candidate, but not the only possibility, since previous mappings may also be involved. To take this into account, when an interpretation is not possible the value of its corresponding mapping is updated as indicated in Table 4.2, according to which constraints were violated. The updates are performed iteratively for each violated constraint. After they are completed, all values are normalized to obtain $\sum_{v \in V_1} \omega(v, v_2) = 1$. In this case, we need to use a normalization method that maintains the values that equal 0.

To find possibly wrong mappings agents use the function μ from Definition 4.11 that provides information about which mappings were made in an interaction. Concretely, we use μ^{-1} to obtain the foreign term that was mapped with a local word. Note that this is only possible under the assumption of bijectivity of τ. When this is not the case, the confidence with which v_1 was a correct interpretation should be $\prod_{v \in \mu^{-1}(v_1)} \omega(v_1, v)$, taking $\mu^{-1}(v_1)$ as a set, instead of $\omega(v_1, \mu^{-1}(v_1))$.

Let us explain the motivation for each update. Violating the negative existential constraint is different to violating a relation constraint, because it expresses a condition over only one message. Therefore if the chosen interpretation violates it, it must be wrong. For this reason, the value of the mapping is set to 0. Similarly, if *before* depends on a message by an agent that has not said anything the mapping is definitely wrong and set to 0.

When a negative relational constraint is violated, there is one message that might have been misinterpreted. We take this into account by subtracting a value that is proportional to the confidence in the mapping of that message with its interpretation. In the case of *premise*, the constraint depends on the immediately previous message. When the constraint requires that it is sent by

Violated Constraint	Update
$!existence(n, \langle a_2, v_1 \rangle)$	$\omega(v_1, v_2) := 0$
$!correlation(\langle a_2, v_1' \rangle, \langle a_2, v_1 \rangle)$ $!response(\langle a_2, v_1' \rangle, \langle a_2, v_1 \rangle)$ $!before(\langle a_2, v_1' \rangle, \langle a_2, v_1 \rangle)$ $!premise(\langle a_2, v_1' \rangle, \langle a_2, v_1 \rangle)$ $!imm\_after(\langle a_2, v_1' \rangle, \langle a_2, v_1 \rangle)$	$\omega(v_1, v_2) := \omega(v_1, v_2) - r \cdot \omega(v_1', \mu^{-1}(v_1'))$
$premise(\langle a, v_1' \rangle, \langle a_2, v_1 \rangle)$	$\omega(v_1, v_2) := 0$ if the last message in I_1 was not uttered by a $\omega(v_1, v_2) := \omega(v_1, v_2) - r \cdot \omega(v_1'', \mu^{-1}(v_1''))$ if $a = a_2$ and $\langle a_2, v_1'' \rangle$ is the last message in I_1
$imm\_after(\langle a_2, v_1' \rangle, \langle a_2, v_1'' \rangle)$ (if $v_1' \neq v_1$)	$\omega(v_1, v_2) := \omega(v_1, v_2) - r \cdot \omega(v_1', \mu^{-1}(v_1'))$
$before(\langle a, v_1' \rangle, \langle a_2, v_1 \rangle)$	$\omega(v_1, v_2) := 0$ if a did not send any message
Other	$\omega(v_1, v_2) := \omega(v_1, v_2) - r \cdot \omega(v_1, v_2)$

Table 4.2: (Reasoning Agent) Updates for each violated constraint, where $r \in [0, 1]$ is a punishment parameter.

an agent different to who actually sent the last message, the mapping being considered is impossible and its value is set to 0. Otherwise, if the message was sent by a_2, the mapping that may have been misinterpreted is the last one, so the update subtracts a value that is proportional to the mapping of the last received word. In the case of *imm_after*, the constraint can be violated if it expected a different word after the last message.

In other cases it is impossible to determine the message that might have been wrongly interpreted. This happens, for example, when *before* is violated (and it does not fall in the case above), when the violated constraint depends on a message that a_1 has sent, or when there is no violated constraint. In these cases agents use the default punishment r.

Lastly, in our experiments we choose $r = \frac{1}{|V_1|}$ because then our technique has the effect of subtracting a larger value when the agent is confident in the correctness of the previous interpretation.

4.4.2 Students and Teachers: Agents that Actively Want to Learn

The agents we considered in the previous section have as their main objective to complete a task, and learning a translation is a byproduct of this process. However, as we mentioned, the interaction I is not an externally observed part of the environment, but is created by the agents themselves. Interlocutors can therefore decide to utter words that will make the interaction possibly more informative in the future, with the particular objective of learning the translation. In the *ordering drinks* example, this is the difference between an agent that wants to drink wine and will only ask for that, and another one that goes to the bar with the main objective of learning the language, and will order anything that will likely lead to an interesting conversation.

To explain the behavior of agents that actively want to learn, let us first define a new class of possible messages, called *predecessors*.

Definition 4.12. A word $v \in V_1$ is *local predecessor* for agent a_1 if there is a relational non-p-necessary constraint in $\mathfrak{P}_1$ that is not *before* or *premise*, and such that the first message of the constraint is $\langle a_1, v\rangle$, and the second one is $\langle a_2, v'\rangle$ for some $v' \in V_1$. The word v is also a predecessor if $!correlation(\langle v', a_2\rangle, \langle v, a_1\rangle) \in \mathfrak{P}_1$ for some $v' \in V_1$

A *foreign predecessor* is defined equivalently, but requiring that the constraint's second message (or possibly the first one for $!correlation$) is a_1. ■

An interaction that includes predecessors will more likely be informative in the future than one that does not. This is because agents learn when constraints are violated, and some relational constraints can only be violated when a predecessor was uttered. We propose two different agents based on how they choose to utter predecessors.

- *Students:* These agents choose local predecessors with higher probability. Concretely, they make unsaid local predecessors twice as likely as the rest of possible messages.

- *Teachers:* These agents choose foreign predecessors with higher probability. Concretely, they make unsaid foreign predecessors twice as likely as the rest of possible messages.

While 'students' try to utter words that will be more informative for themselves in the future, 'teachers' choose those that will be useful for their interlocutors. As we show in the evaluation, 'teachers' make a better use of the available information. This is because foreign predecessors are more useful to learn from than local ones. With local predecessors, the agent that utters a word does not know whether the other agent has interpreted it correctly, but it still makes

inferences about it. Foreign predecessors, instead, are interpreted by the same agent that will later use the rules. If they are interpreted correctly, the inferences drawn from them will also be correct. If they are interpreted incorrectly, no information is obtained. To illustrate this difference, consider an agent a_1 that says v_1 because it has a constraint $!response(\langle a_1, v_1\rangle, \langle a_2, v_1'\rangle)$. After this, it will infer that all the words it receives are not mapped with v_1'. If a_2 interpreted v_1 wrongly and is violating the constraint, these inferences could be incorrect. If instead the constraint is $!response(\langle a_1, v_1\rangle, \langle a_1, v_1'\rangle)$, it will be a_2 who will use the information, but only if it interprets v_1 correctly.

4.5 Experimental Evaluation

In this section we present the experimental results that show the performance of each agent type in different situations. We analyzed their performance in the general case (Experiment 1), when they have external alignments (Experiment 2), when the words in the vocabularies do not appear uniformly in the protocols (Experiment 3), and when only one of the two agents is learning (Experiment 4). The hypotheses that we intend to test with the experimentation are:

1. With repeated interactions agents improve their knowledge about the correct translation.

2. Agents that use the reasoning strategy (Section 4.4.1), as well as those that choose the words they send to learn more (Section 4.4.2), outperform agents following the simple strategy (Section 4.4.1) in terms of the amount of interactions that are necessary to discover the correct translation.

Before presenting the results, let us briefly discuss the generation of the data we used and explain the experimental setup.

Data Generation

The procedure to generate a protocol starts from a vocabulary, and consists in choosing constraints randomly and adding them to the protocol if it remains satisfiable with the new constraint. Since we evaluate techniques that use non-p-necessary constraints, we included only this type of rules. We generated only $!existence$ constraints with $n \leq 2$. We created protocols with different sizes in vocabulary and in the set of constraints. We used the NuSMV model checker (Cimatti et al. 2002) to perform all the necessary satisfiability checks.

Starting from one protocol $\mathfrak{P}_1$ over vocabulary V_1, a compatible protocol $\mathfrak{P}_2$ can be generated in a simple way by taking $\tau(\mathfrak{P}_1)$, where $\tau : V_1 \to V_2$ is a bijective alignment created for a different vocabulary V_2. Note that this implies that all our evaluation is done for bijective translations.

Experimental Setup

The objective of our experiments is to determine how fast agents can learn a correct translation between their vocabulary and the one of their interlocutor. A run of an experiment consists of two agents a_1 and a_2 with vocabularies V_1 and V_2 who are sequentially given pairs of protocols compatible under one same translation τ. The same protocols can appear repeatedly in the sequence, but eventually a new one always appear. We implicitly assume that, after many tasks have been executed, there is only one translation between V_1 and V_2 under which all the pairs of protocols are compatible. Agents interact following each pair of protocols, with the dynamics presented in Section 4.3. To simplify the termination of the interaction, we used bounds that all agents agreed upon to determine the maximum length of an interaction; however, this does not have any effect on the learning process.

We let agents interact 300 times. After each interaction, we measured how close agents were to the correct translation τ. To this aim, we first need to define a measure of how well an agent knows a translation. Since agents always choose the possible mapping with highest weight, we can easily extract from the alignment ω a translation that contains, for each foreign word, the interpretation that would be chosen as a first option in the following interaction. We will refer to the translation extracted from the alignment of agent a_1 as τ_1. The domain of τ_1 are all $v_2 \in V_2$ such that $(v, v_2) \in \mathrm{dom}(\omega)$ for all $v \in V_1$ and $\tau_1(v_2) = \mathit{random}(\mathrm{argmax}_{v \in V_1}\, \omega(v, v_2))$. Note that if there are multiple possibilities with the same value, τ_1 randomly chooses one of them. To compare the local translations τ_1 and τ_2 with the reference τ, we used the F-score measure (see Definition 3.9 in the previous chapter).

We performed experiments parametrized with a protocol and vocabulary size. We initially performed the experiments with the same vocabulary sizes that are used for testing in (Atencia and Schorlemmer 2012), which are $5, 10, 15, 40, 80$. However, the interactions for size 80 were too slow to be able to perform a reasonable amount of repetitions for each agent type. This problem is intrinsic to the protocols we use (since agents have to decide if the messages they want to send are possible), and should be taken into account in future work. Since varying the vocabulary size did not provide particularly interesting results, we show here the experiments for a vocabulary of 12 words and four different protocol sizes. We used the values $r = 0.3$ for the punishment parameter of the simple strategy.

We performed the experiment for six different agents. These are the 'simple' and 'reasoner' agents, and the combination of these with students and teachers: 'student simple', 'student reasoner', 'teacher simple' and 'teacher reasoner'. For simplicity we will sometimes call 'student' and 'teacher' to the first and last two of these combinations, respectively. We only considered pairs of agents of the

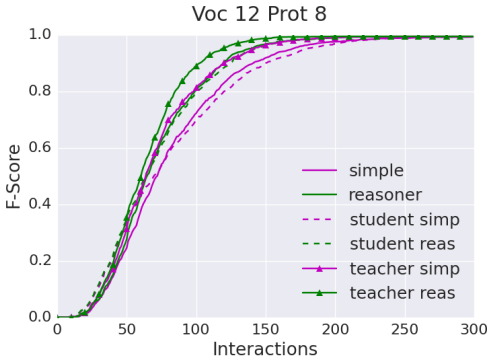


Figure 4.1: F-score curve for different agents

same type, and not their combinations. Each experiment was repeated 10 times, and we averaged the results. The *F-scores* that we refer to in the figures are the average of the *F-score* of each agent that participates in the interaction.

Experiment 1: Convergence

The first experiment analyzes how fast each agent type learns the alignment τ from scratch, when they start with an empty alignment. Figure 4.1 shows the average F-score of the alignment obtained after interacting n times for agents using protocols with 8 constraints. Agents reach an F-score of $\sim 0.8 \pm 0.1$ after approximately 100 interactions. They learn at a particularly fast rate between interactions 25 and 100, when they already have some information but still have to find the correct alignment between some words. 'Reasoner' learns faster than 'simple', and they are both better when combined with 'teacher'. The combination with 'student', on the other side, does not seem to present any improvement. We will discuss this later.

In the figure it is clear how at some point the learning slows down. The most challenging part of the learning process is to find the final mappings to reach the correct alignment, since agents need to wait for informative protocols. To analyze how different agent types perform in this particular moment, we measured how many interactions they have to perform before they find the correct alignment. The plots in Figure 4.2 show how many of the 100 experiments that were performed achieved an F-score of 1.0 after n interactions. We show the results for agents using protocols with 6, 8, 10 and 12 constraints. In this case the difference between the different agents is much more evident. This difference is more pronounced for smaller protocols; when there is less information, it is more evident who makes a better use of it. 'Reasoner' does better than 'simple' because it is able to use the information it obtains after some interactions more efficiently. Agents always improve their performance when combined

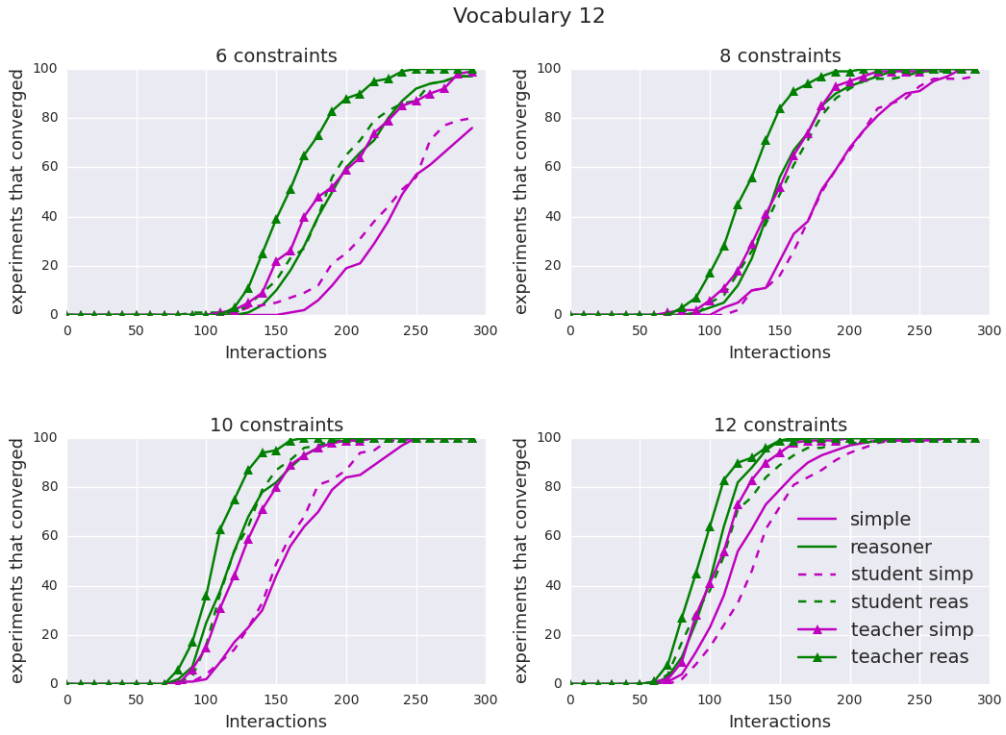
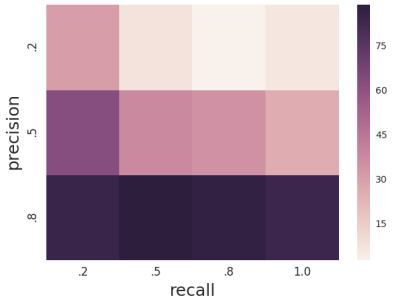


Figure 4.2: (Experiment 1) Performance for different types of agents, for protocols of different sizes (measured in number of rules)

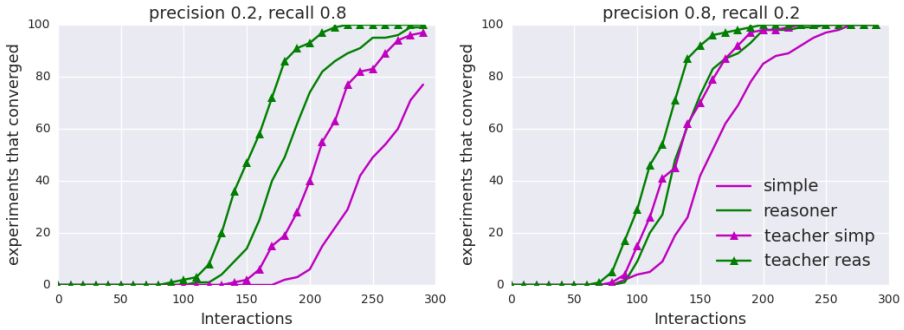
with 'teacher', because this technique use the information in the protocols more efficiently, extracting what is more useful. However, as we have already discussed in Section 4.4.2, 'student' is not consistently better. This is because they learn from local predecessors very often, which depend on how their interlocutor has interpreted messages. However, they have no information about how these words were interpreted. Given that there is no improvement, from now on we will not consider 'student', using only 'teacher'. Since observing how long agents take to converge shows more clearly the difference between agent types, from now on we will show the results in this way.

Experiment 2: Alignment Repair

As we mentioned, interaction-based vocabulary alignment methods can be combined in a simple way with external alignments. Of course, these alignments may have incorrect mappings, in which case our learning method works as a repair technique. We studied the performance of our methods when used by agents with external alignments of different qualities, by varying their precision and



(a) Number of experiments that converged before 200 interactions for different external alignments, for the agent 'simple'.



(b) Results for different agents, with external alignments of two different qualities.

Figure 4.3: (Experiment 2) Results for agents with previous alignments. Vocabulary of 12 words, protocols of 8 constraints.

the recall with respect to τ. We analyzed the combinations between precision and recall levels of $0.2, 0.5$, and 0.8. We also included the recall level of 1.0, to analyze the case in which all correct mappings are found. Again, we performed 100 experiments, with 10 different sets of protocols and 10 repetitions for each set.

Figure 4.3a shows convergence results for 'simple' with different values of precision and recall. Since there are some cases that do not converge in 300 interactions, it is impossible to measure the average amount of interactions that are necessary to converge. Instead, we show how many interactions converged in less than 200 interactions. The figure shows an interesting tendency. While the number of interactions that converge grows with the precision of the existing alignment, the same is not observed for the recall. In fact, agents seem to be worse with larger values of recall. This is because a larger recall value for the same precision means that there are more incorrect mappings. According to

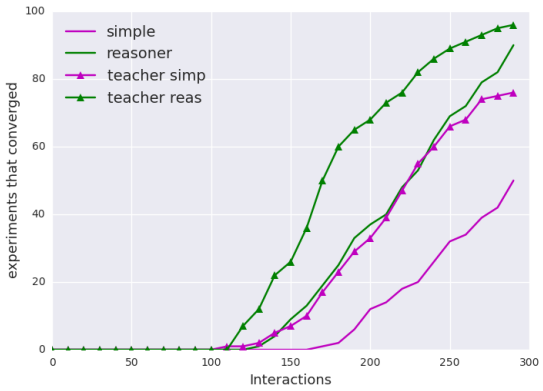


Figure 4.4: (Experiment 3) Results for different types of agents, for the case when $\frac{1}{4}$ of the words are half as frequent as the rest.

the results, incorrect mappings are very harmful for the learning process, and agents perform better when they have less of both correct and incorrect ones.

Another interesting question is how the different agents perform for each alignment quality. Figure 4.3b presents the results for the cases when recall is low and precision is high and vice-versa. Interestingly, we observe that 'reasoner' improves significantly the performance when there is low precision and high recall. This may be because it has a way of discarding alignments completely, which is useful when there are many of them.

Experiment 3: Non-Uniform Word Distributions in the Protocols

Until now, we have always considered that all words are equally likely to be uttered and to appear in the constraints of a protocol. This is not always the case, and words that appear less frequently can be more difficult to learn. To study how this influences our learning technique, we generated protocols in which some words appeared with less likelihood than others. Concretely, we set a frequency for each word v. When building the protocols, each word was chosen to generate constraints with a probability proportional to its frequency. When words appear in few constraints agents have less information about them, making their translation difficult to learn.

Figure 4.4 shows the convergence results for the case of 12 words and a protocol with 8 constraints, where three of the words were half as likely as the other ones to be chosen. Since it is more difficult to find that mapping, agents have a slower convergence when compared to the general case in Figure 4.2. Note that in this case, 'simple' is only able to reach convergence in 300 interactions in half of the cases. It is likely that this is because it is very difficult to obtain

information about the less common words. 'Reasoner' and 'teacher', which use information more efficiently, are much better in this case.

Experiment 4: Unilateral Learning

Having two agents trying to learn each other's language is not a common situation. While it can be useful in the kind of artificial interactions that we presented, in a more realistic scenario there often exists an established language in a community, and it is the responsibility of the agent that does not speak it to try to understand it. The other agent may try to understand the messages it receives, but without making an effort to learn. We tested our methods in this situation by implementing a lazy agent that always chooses a random interpretation between the set of possible ones when it receives a message.

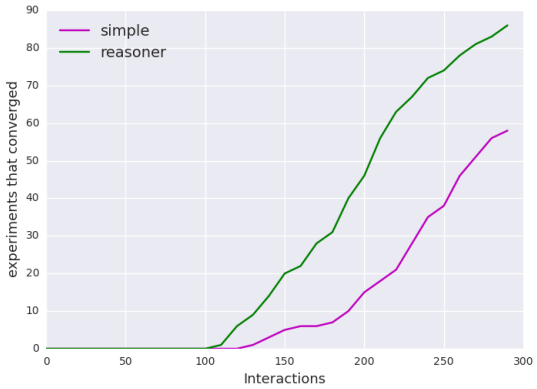


Figure 4.5: (Experiment 4) Results for different types of agents, for pairs where only one agent learns. Vocabulary 12, 8 constraints.

In Figure 4.5 we show the learning process for three types of agents, with vocabulary of 12 words and protocols with 8 constraints. We show results for 'simple' and 'reasoner'. 'Teacher' will not have any effect here, since this agents tries to make its partner learn better, but here only one agent learns. The F-score here is the one corresponding to the learning agent; the lazy partner does not have an alignment. Remarkably, agents (in particular 'reasoner') are still able to learn a translation, even when their interlocutor is not getting any better at interpreting their utterances.

4.6 Including p-Necessary Constraints

Until now, we have presented techniques that exploit the information that can be obtained from constraints that are not p-necessary. As we explained, an agent

can learn from non-p-necessary constraints when considering possible interpretations for a received word, by analyzing which candidate interpretations violate a constraint. Considering only the violation of non-p-necessary constraints is simple because, for many of them, there are only two candidates for the words that can have been interpreted incorrectly. This allows agents to use specific techniques, as does 'reasoner' in Section 4.4.1.

Constraints that are p-necessary, instead, are not affected by new messages; they can only be considered violated when the interaction is over. For example, if $\mathit{existence}(1, \langle a_1, \mathit{buy}\rangle)$ belongs to a protocol, there is always time for a_1 to say *buy* eventually; only when observing a complete interaction one can conclude that the message was never sent.

Even without considering them explicitly, p-necessary constraints provide extra information. A protocol that consists of a mix between p-necessary and non-p-necessary constraints might not be satisfiable even without any violated constraint. Consider, for example, a protocol containing the following set of constraints:

$$\{\mathit{response}(\langle a_1, u\rangle, \langle a_2, v\rangle), !\mathit{correlation}(\langle a_2, v\rangle, \langle a_1, w\rangle), \mathit{existence}(1, \langle a_1, w\rangle)\}$$

In this case, the message $\langle a_1, u\rangle$ cannot be uttered, since then a_2 would have to say v, making it impossible for a_1 to say w.

Protocols that include p-necessary constraints already take them into account in the simple update we proposed in the first section, since the unsatisfiability of subsets of constraints is considered in the definition of *possible messages*. For example, according to our definition $\langle a_1, u\rangle$ is never a possible message for the protocol above. This effect is more pronounced if the protocols of both agents have extra information specifying that an interaction should finish after a certain number of utterances. We call these protocols *bounded*.

Definition 4.13. Let M be a set of messages. A bounded protocol over M is a tuple $\langle \mathfrak{P}, n\rangle$ where $\mathfrak{P} \subseteq \mathit{Cons}(M)$, and n is a positive natural number.

The models of a bounded protocol $\langle \mathfrak{P}, n\rangle$ are interactions I over M such that $\mathit{len}(I) \leq n$ and $I \models \mathfrak{P}$. We require that there exists at least one such interaction. Two protocols $\langle \mathfrak{P}_1, n_1\rangle$, $\langle \mathfrak{P}_2, n_2\rangle$ are compatible if $\mathfrak{P}_1$ and $\mathfrak{P}'_2$ are compatible, and $n_1 = n_2$. ■

For bounded protocols, the definition of possible messages, which remains the same, requires that the interaction can be completed satisfying the protocol in a maximum number of steps, which is more informative.

In Table 4.3 we show, for different types of protocols, the average of how many interactions agents need to perform before converging to the correct translation. We use a vocabulary of 8 words and different protocol sizes. We compared the case of non-bounded protocols only with constraints that are not

p-necessary (such as the ones in the previous section), and of both bounded and non-bounded protocols with a mix of p-necessary and non-p-necessary constraints. We also included an agent that has the same mixed protocols, but only takes into account their non-p-necessary constraints, to show that they do play a role in inferring the alignment. In all cases we used the 'simple' agent.

	Non-p-necessities	Mixed	Mixed, agents using only non-p-necessities	Mixed, bounded
Prot 6	117.72	180.88	215.82	96.68
Prot 8	87.92	116.64	149.26	71.92
Prot 10	69.72	82.78	125.9	61.1

Table 4.3: Average number of interactions before convergence, for the 'simple' agent, with vocabularies of 8 words.

Taking into account p-necessary constraints explicitly is more difficult. We explored two possibilities. The first one consists in analyzing the violated p-necessary constraints at the end of an interaction. This can be done in different ways; either punishing the chosen mappings or rewarding those that would satisfy the violated constraints. However, both methods deal with highly uncertain information. For example, consider a simple case in which a_1 finds that $\mathit{existence}(1, \langle a_2, v \rangle)$ was violated in an interaction. First, a_1 cannot know if the interaction finished successfully for a_2, or if it finished because there was no possible message. Second, the available information only allows a_1 to either punish all mappings made or to reward the mapping of each received word with v. Since only one of those words needs to be v, this implies all the rest are wrong updates. The second approach does not use violated p-necessary constraints, but those that were satisfied. It consists on rewarding those mappings that made a p-necessary constraint become valid. In the case before, if w was mapped with v, the agent would reward this mapping. However, this also resulted in rewarding many wrong mappings.

Due to the high uncertainty, none of these approaches resulted in significant improvements in the performance. In some cases they were even worse than the basic case. It remains as future work to develop useful techniques for p-necessary constraints, or to analyze the case in which the protocols consist of only constraints of this kind.

4.7 A Purely Logical Approach to Alignment Inference from Interactions

From a technical perspective, the work we presented in the past sections is a way of using learning techniques to extract information from logical specifications.

There exist other inference methods to obtain this kind of information, some of which have been extensively studied, that are not based on probabilistic techniques. Instead, these methods use the logical information directly to make deductions. In this section we introduce an ad-hoc logical agent that extracts all the available information from the interactions it is involved in. This method extracts more precise information from the experience, since it can take into account the exact information about the relations that it sees. However, it has no way of extracting information from an uncertain situation, which can only be considered with a probabilistic approach. For example, it can learn nothing at all when a violated constraint involves a word interpreted by its interlocutor.

We will call 'logical' to the agent that uses only logical inferences, that works as follow. First, it generates all possible translations. Since it does not know the foreign vocabulary, it represents possible translations as all permutations of its own vocabulary, which will be mapped with a unique sequence of foreign words as soon as the agent learns them. At the beginning of each interaction it chooses a translation between the possible ones to translate received words. Then, it starts interacting with the other agent. When it is its turn to speak, 'logical' chooses the utterance in the same way as 'simple'. Interpretations are chosen according to the selected translation. When a message is received, it analyzes the possible translations and discards those that are impossible according to this new evidence. This is done in a similar way to how 'reasoner' updates its alignment, since it is based on the same information. Table 4.4 shows the actions for each case. If the chosen translation is updated as impossible, agents finish the interaction.

To evaluate the performance of 'logical' we performed the same kind of experiments as before, letting agents interact for 500 interactions. We used a vocabulary of 8 words and protocols with both p-necessary and non-p-necessary constraints. As before, we measure how many interactions agents perform until they both find the correct translation, that is, when their set of possible translations has only one element. We compared this value to the number of interactions after which 'simple' and 'reasoner' obtain an F-score of 1. As before, we averaged the values for 100 experiments, performing 10 repetitions for 10 different protocol sets.

Table 4.5 shows the convergence results for different agents and protocol sizes. 'Logical' takes more to converge than 'simple', even when the latter one uses no specific information about the type of constraints. When comparing 'logical' to 'reasoner', which uses the same information, we can observe that 'reasoner' is significantly faster.

The time of convergence is, however, not the main problem of 'logical'. The vocabularies of 8 words that we used in the experiments were the larger ones for which the technique could be executed on our server. After that, it became prohibitively space-consuming, and was automatically killed. In our learning

Violated Constraint	Update
$!existence(n, \langle a_2, v_1 \rangle)$	discard translations τ' if $\tau'(v_2) = v_1$
$!correlation(\langle a_2, v_1' \rangle, \langle a_2, v_1 \rangle)$ $!response(\langle a_2, v_1' \rangle, \langle a_2, v_1 \rangle)$ $!before(\langle a_2, v_1' \rangle, \langle a_2, v_1 \rangle)$ $!premise(\langle a_2, v_1' \rangle, \langle a_2, v_1 \rangle)$ $!imm\_after(\langle a_2, v_1' \rangle, \langle a_2, v_1 \rangle)$	discard translations τ' if $\tau'(v_2) = v_1$ and $\tau(\mu^{-1}(v_1')) = v_1'$
$premise(\langle a, v_1' \rangle, \langle a_2, v_1 \rangle)$	discard translations τ' if $\tau'(v_2) = v_1$ if the last message in I_1 was not uttered by a --- discard translations τ' if $\tau'(v_2) = v_1$ and $\tau(\mu^{-1}(v_1')) = v_1'$ if $\langle a_2, v_1'' \rangle$ is the last message in I_1
$imm\_after(\langle a_2, v_1' \rangle, \langle a_2, v_1'' \rangle)$	discard translations τ' if $\tau'(v_2) = v_1$ and $\tau(\mu^{-1}(v_1')) = v_1'$
$before(\langle a, v_1' \rangle, \langle a_2, v_1 \rangle)$	discard translations τ' if $\tau'(v_2) = v_1$ if a did not send any message
Other	nothing

Table 4.4: Updates for each violated constraint.

	'simple'	'reasoner'	'logical'
6 constraints	215.82	164.14	310.82
8 constraints	149.26	112.4	203.5
10 constraints	125.9	96.1	157.84

Table 4.5: Convergence for vocabularies of size 8

techniques, the number of possible mappings in ω grows polynomially with the size of the vocabulary (n^2). The number of possible bijective translations grows much faster, proportionally to the factorial ($n!$). In addition, while the mappings are two numbers, the translations are n mappings in themselves. This makes the logical technique use much more memory, and also much more computing time since the techniques require to perform searches through all the space. The logical agent can be optimized to use less space, but learning agents do it automatically and converge faster.

In addition, learning agents achieve partially correct translations while learning, which means that they can start interacting successfully earlier, even when they still have not discovered the complete translation.

Another important advantage of probabilistic learning techniques is that they can incorporate possibly erroneous previous data. As we discussed in earlier sections, our methods allow to consider external alignments even when not all their mappings are correct, and are able to repair those wrong correspondences. A logical approach, on the other side, has no way of considering possibly wrong mappings. For similar reasons, the agents that we described in previous sections are able to deal with some errors in the transmission. To this end, it is only necessary that agents do not assign 0 to the confidence on any mapping, or that they normalize using a method that leaves all the values in the open interval $(0, 1)$. 'Logical', instead, cannot handle this possible erroneous evidence in a simple way.

Finally, 'logical' cannot deal with non-bijective translations as easily as the agents that use our techniques. Consider a situation in which an interpretation violates, for example, a constraint $!response(v_1, v_1')$, and the agent has mapped both v_2 and v_2' with v_1. The logical agent does not know which of these mappings was incorrect, if any, and therefore it cannot learn anything from this situation. As we mentioned before, 'reasoner' can easily incorporate this uncertainty by considering the product of the confidences of both mappings.

4.8 Vocabulary Alignment for Agents with Flexible Protocols

An important restriction that we imposed on agents until now is that their protocols are compatible. This was a useful decision to focus our work on how agents can learn alignments from shared interaction specifications. However, in practice this is a strong restriction, since it is unlikely that agents will have exactly the same notion of how tasks are performed. To give an example, the procedure to order coffee, that we used as an example in Chapter 1, can be significantly different even between European countries, where different questions or requirements can be expected. As put by Umberto Eco,

> "The words coffee, cafe, and caffe can be considered as reasonable synonyms when they refer to a certain plant. Nevertheless, the expressions 'donnez-moi un cafe', 'give me a coffee', and 'mi dia un caffe' (certainly linguistically equivalent to one another [...]) are not culturally equivalent. Uttered in different countries, they produce different effects and they are used to refer to different habits. They produce different stories." (Eco 2008, p.18)

This raises the question of what agents can learn if the protocol is not completely shared. Our answer, until now, was that if the differences are small, they will be ignored by the learning methods, which work statistically. If, instead, there are significant differences, agents have nothing to learn, since there is no translation that is useful to perform the tasks together. In the extreme, if protocols are completely different there is no joint task to talk about. However, so far we have not performed a systematic analysis of how different is too different or, more generally, of how not sharing the complete structure affects the learning process.

We now propose an approach that considers more carefully the question about the extent to which agents can align their vocabularies when their protocols are different. To this aim, we introduce a new version of ConDec protocols, in which each constraint has a weight that represents a punishment received when that constraint is violated. This punishment can be interpreted in different ways, among which we mention two:

1. As a way of expressing preferences over different flows of the interaction: constraints with heavier weight represent those that agents prefer not to violate.

2. As a way of expressing degrees of confidence on constraints: when there is uncertainty about the interaction context, weights can represent how sure an agent is about a certain rule.

The following is a formal definition of flexible protocols.

Definition 4.14. Consider again a vocabulary V and a set of agent IDs A, and a set of messages $M = V \times A$. A *flexible protocol* over M is a set $\mathfrak{P}^f \subseteq Cons(M) \times [0, 1]$, where each constraint is associated with a value. For simplicity, we will say that a constraint $c \in Cons(M)$ belongs to $\mathfrak{P}^f$ if it is in a tuple in the protocol, and we will refer with $\rho(\mathfrak{P}^f, c)$ to the weight of constraint c in $\mathfrak{P}^f$. ■

As an example, consider again the *ordering drinks* example that we used in the previous sections. Assume a waiter and a customer that have the same constraints as in the protocols in Section 4.2.2, that were not compatible. Now, however, agents have flexible protocols. The first six constraints, which are shared, have high weight. The waiter has an extra constraint saying that the customer should not order two different alcoholic beverages in one interaction. This constraint, however, is less strict than the others, since the waiter is willing to accept that behavior some times. The protocols would be specified as follows.

$$\begin{aligned}\mathfrak{P}_C = \{&\langle existence(1, \langle W, \mathsf{to\ drink}\rangle), 1\rangle,\\ &\langle premise(\langle W, \mathsf{to\ drink}\rangle, \langle C, \mathsf{beer}\rangle), 1\rangle,\\ &\langle premise(\langle W, \mathsf{to\ drink}\rangle, \langle C, \mathsf{wine}\rangle), 1\rangle,\end{aligned}$$

$$\langle response(\langle C, \mathsf{beer}\rangle, \langle W, \mathsf{size}\rangle), 1\rangle,$$
$$\langle premise(\langle W, \mathsf{size}\rangle, \langle C, \mathsf{half\ pint}\rangle), 1\rangle,$$
$$\langle premise(\langle W, \mathsf{size}\rangle, \langle C, \mathsf{pint}\rangle), 1\rangle\}$$

$$\mathfrak{P}_W = \{\langle existence(1, \langle W, \mathsf{da\ bere}\rangle), 1\rangle,$$
$$\langle premise(\langle W, \mathsf{da\ bere}\rangle, \langle C, \mathsf{birra}\rangle), 1\rangle,$$
$$\langle premise(\langle W, \mathsf{da\ bere}\rangle, \langle C, \mathsf{vino}\rangle), 1\rangle,$$
$$\langle response(\langle C, \mathsf{birra}\rangle, \langle W, \mathsf{tipo}\rangle), 1\rangle,$$
$$\langle premise(\langle W, \mathsf{tipo}\rangle, \langle C, \mathsf{piccola}\rangle), 1\rangle,$$
$$\langle premise(\langle W, \mathsf{tipo}\rangle, \langle C, \mathsf{media}\rangle), 1\rangle,$$
$$\langle !correlation(\langle C, \mathsf{birra}\rangle, \langle W, \mathsf{vino}\rangle), 0.5\rangle\}$$

4.8.1 Updating Technique

In previous sections, which dealt with non-flexible protocols, the objective of the learning process was to find a correct translation τ under which the protocols were compatible. The notion of compatibility cannot be applied to flexible protocols, since the question of whether an interaction is possible or not is not meaningful anymore. When interacting with flexible protocols, any sequence of messages can be accepted, but at some cost. This implies that there is no reference translation τ that agents can use to understand each other completely. Instead, translations are associated with the expected punishments agents will receive when interacting.

The goal of the technique that we present here is to find a translation that minimizes the punishment that agents receive while interacting. To do so, it is necessary to take into account the weight of the rules that would be violated for each interpretation. Consider the same interacting situation as before, but with agents a_1 and a_2 using flexible protocols $\mathfrak{P}_1^f$ and $\mathfrak{P}_2^f$ respectively. Again, each agent has an alignment ω that is updated when choosing interpretations, which is initialized as before. If a_1 receives a word v_2 from a_2 after interaction I, it will consider the weight of all the constraints that are violated by choosing each interpretation, and update ω accordingly. For a particular interaction I, protocol $\mathfrak{P}_1^f$, and a word $v \in V_1$, let us refer with $Viol(v)$ to all the constraints $c \in \mathfrak{P}_1^f$ that are violated by $I \,.\, \langle a_2, v\rangle$. The update for each $v \in V_1$ does the following iteratively for each $c \in Viol(v)$:

$$\omega(v, v_2) := \omega(v, v_2) * (1 - \rho(\mathfrak{P}_1^f, c))$$

After the updates, the values are normalized to make $\sum_{v \in V_1} \omega(v, v_2) = 1$. The alignment ω is used in the same way as before to obtain a local translation, which agents can use to decide how to interpret foreign messages.

4.8.2 Experimental Evaluation

Flexible protocols, along with the updating technique that we proposed, can be analyzed from two points of view. First, we can consider agents that have structurally different protocols for the same task, without requiring any a priori relation between them. In this case, the technique can be used, as we discussed, as a tool to find a translation that minimizes the punishment received by agents for that specific protocol, or set of protocols.

A second way of considering flexible protocols is as local variations of a common compatible part. That is, we consider agents with protocols that have a majority of shared constraints, but also have some others on which they do not agree. An example of this situation is the *ordering drinks* example that we proposed, where the waiter has only one extra constraint. In this case we study whether agents can use the technique to find a correct translation (that is useful for the common part) despite the differences. In the following sections we explain the experiments that we performed to study these two cases.

Learning a Translation that Minimizes Punishment

In the first experiment we study the application of our technique when agents have different protocols, without making any similarity assumption. The objective of the interlocutors is to find a translation that would make them pay as little as possible. For each experiment we generated a pair of flexible protocols, made agents interact with them repeatedly using the update technique described above, and then measured the punishment that they received when using that alignment as a translation.

These experiments are a simple study with a small vocabulary of 4 words. We built the protocols by specifying a number of rules and a total weight to be distributed among them. Concretely, we used protocols with 6 and 8 rules, and total weight 3.0 and 5.0. In each experiment both protocols have the same number of constraints and the same added weight. In a run, we let agents using two vocabularies go through a learning phase in which they interacted n times, performing the experiment for $n = [0, 2, 5, 10, 15, 20, 30]$. During these interactions they updated their alignments ω. After this training phase, we ran a test phase of 20 interactions, during which agents did not perform any update and interpreted foreign words using the alignments obtained in the training phase. We measured the punishment received by each agent as the sum of the punishment received on each interaction in the test phase. We repeated the experimentation 10 times, for 10 different protocols. Figure 4.6 shows the results obtained for each experiment. We can see that with more training, agents find translations that allow them to interact receiving less punishment.

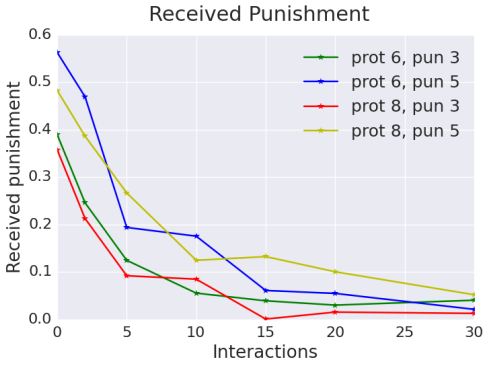


Figure 4.6: Learning a translation that minimizes punishment. Results for a vocabulary of 4 words.

Learning a Correct Translation in Protocols with Variations

In the second type of experiments we were interested in analyzing the learning techniques when applied by agents with protocols that diverge by some degree from a shared compatible part. To generate these kind of protocols, we used four parameters:

- r_s : the number of shared constraints
- r_d : the number of different constraints
- w_s : the total weight for shared constraints
- w_d : the total weight for different constraints

We started from two protocols over vocabularies V_1 and V_2, each with r_s constraints, that were compatible under a translation τ. We distributed the weight w_s over those constraints, that constitute the shared part of the protocols (to simplify the experimentation, we did this uniformly). Then, we added r_d different constraints to each protocol, and distributed the weight w_d between them (not necessarily uniformly).

In these experiments, similarly to the ones in Section 4.5, we let agents interact 140 times and measured when (and if) they converged to τ. However, there is an important difference with respect to the experiments in previous sections. Before, it was enough to consider that agents had found τ when their learned translations were equivalent to it since, in non-flexible protocols, using τ always leads to correct interactions. This is not the case anymore, since agents could receive punishments even using τ, that could made them change

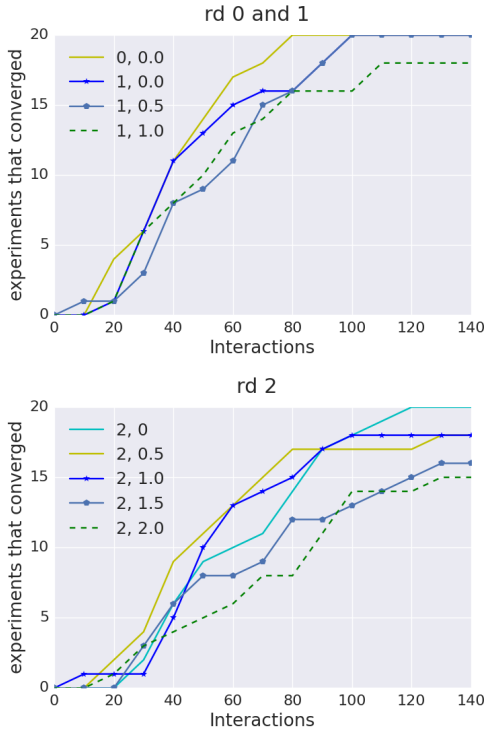


Figure 4.7: Learning a correct translation in protocols with variations. Results for a vocabulary of 4 words and protocols with 8 common constraints. The first number in the caption corresponds to the value of r_d and the second one to that of w_d.

the translation. For this reason we considered an agent had converged after they had spent 15 interactions without changing the translation (but we recorded the number where they first got to τ, before those 15 interactions).

In our experiments, we considered a large set of shared constraints ($r_s = 8$, and $w_s = 6.4$ divided in 0.8 for each constraint), and studied the convergence to τ for different values of r_d and w_d. On each run, agents perform 20 interactions with different pairs of protocols. Figure 4.7 shows the number of runs that converged after each amount of interactions. We used $r_d = \{0, 1, 2\}$ and $w_d = \{0.0, 0.5, 1.0, 1.5, 2.0\}$. Some combinations were not performed because the weight was impossible to distribute over the amount of rules. Unsurprisingly, we see how more and heavier different constraints imply τ is more difficult to find. We also observe an unexpected effect. Even with the same weight, more constraints slow down the convergence. We think this is because agents still choose which word to utter by checking that no constraints are violated. Therefore, a

larger number of constraints means that more interactions could finish in failure. Another thing to comment on is that the difference between small weights cannot be seen correctly in this experiments (for example, compare $r_d = 2, w_d = 0.0$ to $r_d = 2, w_d = 0.5$). We plan to study this more exhaustively in the future, with a larger set of experiments.

To have a deeper comprehension of how these techniques work, they should be compared to the performance of the original learning methods for non-flexible protocols, when these are used for protocols that have some diverging constraints. In this way, we would be able to detect if taking into account the weight of different constraints explicitly is better than considering them as noise.

4.8.3 Discussion

This section shows that our techniques can be used even when the interaction protocols of the interlocutors are not completely compatible. Interestingly, agents can use them to learn a translation that minimizes the punishment they obtain when interacting. This extension takes our understanding of correct translations a step further. If before a correct translation was one that allowed agents to interact correctly, now it is one that makes them pay less. This is a dynamic notion that makes almost no assumption on the agents, and can therefore be applied in really open situations.

This work is still in a preliminary state, but we see it as a basis to many extensions. First of all, the experimental part needs to be improved, considering larger protocols and making a more accurate analysis. The current experiments should only be seen as tests that give an idea of how the method works and provide input for future experimentation. For example, in the first experiment it is possible to consider the optimality of the solution the agents find. That is, if they actually converge to the alignment that makes them pay less. This would be useful to understand how the technique works. In addition to this, there are different questions to consider regarding the technique. We think it would be particularly interesting to consider interactions between agents with different kinds of preferences. For example, what kind of translation would a pair of agents reach if one of them has very high preference values and the other one very low?

Finally, something that we did not develop in this section is how the similarity between a pair of flexible protocols can be measured. In Chocron and Schorlemmer (2017a) we propose to consider a measure of the *expected punishment* that agents would receive when interacting with a pair of translations and protocols. We show that, to compute it, it is enough to consider all interactions of length two, since all possible constraints violations are represented in those.

4.9 Conclusions

In this chapter we developed techniques that allow agents to find vocabulary alignments when using open protocols to interact. We made a number of assumptions, namely requiring agents to share the same models for each task and considering the existence of a bijective translation. These assumptions can be relaxed, but at the cost of a slower learning process, since an extra component of uncertainty is added. Although our methods converge slowly, they use only very general information, and can be easily combined with other methods that provide more information about possible mappings, as we show with the integration of previous translations.

Open, constraint-based protocols need a different type of learning technique than transition-based ones. This is because before the information they give is different, since they do not mention explicitly everything that can be said but instead specify constraints that restrict possible utterances. On one side, the learning process for open protocols has the disadvantage that there is less information on each state (since many more messages are possible). In addition, to decide whether a message is possible it is necessary to employ a model checker, which can be slow. On the other side, the problem of *delayed reward* we dealt with in Chapter 3 is no longer here, since agents can find immediately if a constraint is violated. When this happens, the constraint depends only on up to two words, which makes possible the development of *smarter* techniques such as the 'reasoner'.

There exist many possible directions of research derived from this work. First, we could relax the assumption that agents do not share any meta-language, considering agents that can in some way exchange information about the interaction. For example, considering only that agents can communicate whether they finished a task successfully would make it possible to reason about p-necessary constraints when an interaction ends. An approach like this would relate our work to the one in (Santos et al. 2016a) about dialogs for meaning negotiation. An aspect that should be improved in future work is the performance in terms of runtime per interaction, since it was very slow for larger vocabularies.

Chapter 5

Inference of Commitment Semantics

In the previous two chapters we studied how agents can align their vocabularies when they use specifications based on finite state machines and logical constraints. Although in different ways, these two formalisms define rules about what can and what cannot be said at different points in the interaction. A different approach, described in the introduction as *commitment semantics*, goes a step further, by providing agents with ways of manipulating these rules.

In this chapter we explore how commitment semantics can be learned from the experience of interacting. Unlike in the previous chapters, we do not propose an alignment technique, but a method that allows external agents to infer a semantics from observing interactions. As before, we evaluate the results with simulations over specifications generated randomly.

5.1 Introduction

Commitment semantics were first proposed by Singh (1999) with the objective of providing a tool to specify interactions that would overcome the weaknesses of existent specification languages. On one side, as we discussed in Chapter 1, methods based on mental states lacked a rigorous semantics and therefore yielded specifications that were difficult to verify. On the other side, protocols described with automatas or constraints were prone to be over-specified, since these methods establish rules about the flow of the interaction that must be always followed.

Instead of determining rules, commitment specifications define the meaning of words in terms of their social effects. Concretely, meaning is described in terms of operations over commitments. A commitment is a relation between a debtor and a creditor, where the first commits to do something if the creditor

does something else. For example, a specification could determine that when an agent says Offer, it is creating a commitment to make an item available if the receiver of the message pays for it. In more formal terms this would be expressed by saying:

$$\textsf{Offer} \textbf{ means } \mathit{Create}(\mathit{pay}, \mathit{item})$$

It is important to observe that uttering *Offer* implies the creation of the commitment, but is not a commitment in itself. The meaning of utterances is expressed in terms of operations over commitments, which, in a way, gives interlocutors the freedom to set the rules. Apart from *Create*, there exist other operations to, for example, delete or change commitments. Rules, that were external in automata- or constraint-based specifications, are now first-class elements that agents can manipulate.

A commitment specification consists of a set of definitions, like the one above, that determine the social meaning of words. Naturally, a commitment specification is only useful if it is shared by all the interlocutors. If one of the agents has the specification above for Offer, while another one thinks it is enough to acknowledge the offer to obtain the item (Offer **means** $\mathit{Create}(\mathit{ACK}, \mathit{item})$) , it is unlikely that they will have a successful transaction. The community working on commitment semantics has already considered different aspects of This interoperability problem. For example, Chopra and Singh (2008) proposed a definition of interoperability for commitments that they call *constitutive interoperability*. Simply put, two agents can interoperate if they share the same commitments. This work is extended in (Chopra and Singh 2015), where the authors provided solutions to commitment misalignment, which can occur in an asynchronous environment where messages can be lost or delayed. Other relevant approaches are the work by Günay et al. (2015) tackled the problem of generating protocols dynamically, adapting to an open situation, and by Ajmeri et al. (2016), who presented a technique to solve conflicting commitments online.

In this chapter we study the problem of semantic inference for agents using commitment specifications. We consider a *new kid in town* situation: a foreign agent that arrives to a community whose commitment specification it does not know. There is no shared meta-language, so the agent cannot ask directly what words mean. However, it can observe interactions between other agents, who do know the community's vocabulary and its social semantics. Here we analyze under which conditions the foreign agent can infer the commitment specification with only this information. Importantly, the meaning of a word is not directly observable, but has to be inferred by analyzing repeated behavior in different interactions.

Note that this problem is different from the ones that we approached in previous chapters. Until now we assumed that the specification was known by both

agents, and they used it to find an alignment between their vocabularies. Now, instead, the semantics are not known, and an agent learns them by observation. In addition, now the learner is an external agent, when before it the interlocutors learned. The two problems are important aspects of interoperability.

As before, we use a simple probabilistic technique in which the learning agent infers possible meanings from the interactions it observes. We show how, when using a basic commitment semantics, agents can learn a specification that is useful to interact, but which is not necessarily the correct one. Incorporating simple semantic extensions is enough to infer the actual specification. To the best of our knowledge, the problem of commitment semantics inference is, until now, unexplored.

The problem of inferring commitment semantics is closely related to the more investigated problem of *norm inference*, that studies how an agent can learn what actions are allowed in a community and which ones are not (Cranefield et al. 2016). This similarity is unsurprising given the relation between the notions of *obligation* norms and commitments, discussed by Singh (2012). In the same paper, however, the author points at critical differences. Mainly, agents cannot operate with obligations in the same way they can cancel or release commitments. The problem of inferring rules therefore lacks the complexity that, as we will see, arises from these operations. Also related is the problem of *process mining* (van der Aalst 2011), that provides techniques to analyze business processes from the information that is stored in event logs. In this case, procedures are specified with automatas or rules. Our work can be used in a similar way when the specification has a component that can be expressed with commitments.

The rest of this chapter is organized as follows. In the following section we describe possible scenarios in which the techniques in this chapter would be useful. In Section 5.3 we formally describe the language to specify commitments. Section 5.4 presents a probabilistic technique to discover social meanings from observing interactions between two agents. This problem is particularly challenging because the possibility of operating with commitments adds significant uncertainty, since rules can be created and deleted at any time. Section 5.5 experimentally shows that the basic commitment semantics does not allow to infer meaning correctly. Section 5.6 discusses different extensions that enable agents to learn the specification.

5.2 Scenarios

The general situation that we consider in this chapter can be described as follows. We assume there exists an established community with a shared commitment specification that guides interactions between their members. This specifica-

tion defines which utterances trigger which operations over commitments. To interact successfully, agents must know this specification. Otherwise they may unintentionally engage in commitments without complying with them later. We consider a foreign agent that wants to interact with agents in the community, but which does not have access to the specification. This can happen for different reasons:

The foreign agent is unable to ask for it. If the agent knows nothing about the foreign language, it may not even be able to ask someone else for the specification.

The foreign agent is unable to understand a specification. Even if the specification is available, it may be impossible to understand for the new agent. This is because the words or symbols used to describe meanings can be different from the ones the agent would use. To make an analogy, this would be like giving a dictionary with definitions in English to somebody who does not speak any English.

There is no specification. The specification may not exist as something that can be shared. Social conventions are dynamic and fast-evolving, and they are not necessarily always explicitly written down. The fact that agents can follow a convention does not necessarily mean that they can make it explicit.

The technique that we propose allows the foreign agent to infer the commitment semantics used by the community only by observing interactions between other participants who already know it. In this way, the agent can start interacting only when it is sure that it knows the semantics (because all the interactions it sees adjust well to the specification it inferred). We do require that the agent can observe many interactions, particularly when the vocabularies are large. We identify three types of scenarios where this is possible.

Scenarios with many available interactions. This includes any kind of open environment with many participants and public interactions. An example are the auction or market systems described by d'Inverno et al. (2012). An agent who wants to start participating in an auction community has, in addition to the reasons mentioned before, an extra incentive to avoid asking for definitions: in such a competitive environment, it could be deceived. Observing public interactions between other agents is a way of learning the semantics discretely and efficiently.

Scenarios in which a log is available. In this case an agent tries to enter a smaller, closed community with well-defined internal semantics. An example of such a community are the groups of parents described by Koster et al. (2012), where members interact with each other to collaborate in everyday tasks. If the community gives the agent access to a log of interactions, it could learn the semantics before starting to participate.

Scenarios with emergent semantics. An agent that already knows the language of a community can use the techniques that we propose to understand new meaning that is created dynamically by other interlocutors. This kind of meaning is known as *emergent semantics* (Aberer et al. 2003) and it is difficult to capture since it is usually not explicit. Using our techniques an artificial agent could infer, for example, the commitment meaning of expressions that appear spontaneously in social media communities only by analyzing their use. Since the agent only needs to learn the meaning of a small vocabulary, it does not require that many interactions.

5.3 Commitment Specifications

A semantic based on commitments describes the social effect that utterances have without imposing any restriction on how the flow of the interaction will be. A commitment is an abstract concept relating two agents (debtor and creditor) and two propositions (antecedent and consequent). The intuitive meaning of a commitment is that, when it holds, the debtor is obliged to bring about the consequent if the creditor enforces the antecedent. As an example, the commitment created by *Offer* that we mentioned in the introduction could have agent a_1 as a debtor, a_2 as a creditor, and the actions of paying and delivering an item as antecedent and consequent, respectively. Agents can operate with commitments, creating, deleting, or even assigning them to other agents; they do this by sending to each other messages that are associated with the operations.

The syntax and semantics of (operations over) commitments have been formalized in multiple ways. Originally, commitments were defined over a propositional language, and the antecedent and consequent were propositions on this language. A commitment implied that a debtor would make the consequent true if the creditor made the antecedent true. A separated vocabulary was used to communicate the operations. We will, instead, follow the approach proposed in (Marengo et al. 2011). In this work, commitments are defined over the same events that trigger them. That means that an agent can, by performing an action u, commit to perform v if the creditor performs w. We will use this idea considering events to be messages that are exchanged between agents. In this way, agents operate over commitments by sending messages, and at the same

time commitments are defined over the utterance of words. That is, a commitment with v as antecedent and u as consequent means that the creditor must say u if the debtor says v. In the example above, the creation of the commitment to buy an item is triggered by uttering the word *Offer*, and it says that, if the debtor says *Pay*, the creditor must say *Item* later. In this way, we provide a general approach by keeping all the specification at the level of utterances, without having to deal with how agents perceive the truth-value of a proposition.

In this section we describe the syntax and the semantics of our language to specify commitments. While the idea is heavily inspired by the work in (Marengo et al. 2011), we present it in a different way that will ease the exposition of the inference techniques. We also made some simplifications that make the inference more approachable. Concretely, we allow words to have only one meaning and operations over commitments are expressed over single messages. We also consider interactions between two agents only.

5.3.1 Syntax

A *commitment specification* relates words in a vocabulary with their social meaning, that is, with operations over commitments. We consider three of the basic commitment operators: *Create*, *Cancel*, and *Release* (Venkatraman and Singh 1999). Intuitively, *Create* initiates a commitment contract in which a debtor commits to say something (the consequent) if the creditor says something else (the antecedent). Both *Cancel* and *Release* are ways of finishing the commitment without necessarily uttering the consequent. The difference between them is that *Cancel* is uttered by the debtor, while only the creditor can *Release* a commitment. Since we only consider interactions between two agents, we do not include the *Assign* and *Delegate* operator that only make sense in larger communities. From now on, we use a vocabulary V and a set of agent IDs $A = \{a_1, a_2\}$.

Definition 5.1. Let $v, v' \in V$ be two words. A *commitment operation* is a term $op(v, v')$ where $op \in \{Create, Release, Cancel\}$. We call O_V the possible commitment operations over words in vocabulary V. ■

A specification is a function between V and the set of commitment operations. Not all words in the vocabulary need to have a commitment meaning, so we include a *None* term to the co-domain of the function.

Definition 5.2. A *specification* over V is a total function $\mathbf{means} : V \to O_V \cup \{None\}$. ■

The **means** function relates words with their social meaning only, letting us focus on the inference of that kind of semantics in particular. Of course, words in the vocabulary could have additional meanings. For example, there

could exist a physical dimension of semantics that relates words with events, in which a word Pay corresponds to a transference to a bank account. The problem of learning correspondences between words and an observable physical meaning has been extensively studied (for example see (Siskind 1996), discussed in Chapter 2, for a cognitive approach) and we do not consider it here.

Example. We will use the example we already mentioned to illustrate concepts throughout this chapter. From now on, we will refer to it as the *transaction* example. Consider a vocabulary $V = \{$Offer,Withdraw, Complain, Pay, Receipt, Item, ReturnMoney, Reject$\}$. The following one is a possible commitment specification:

$$\begin{aligned}
\textbf{means}(\mathsf{Offer}) &= \mathit{Create}(\mathsf{Pay}, \mathsf{Item}) \\
\textbf{means}(\mathsf{Item}) &= \mathit{Create}(\mathsf{Complain}, \mathsf{ReturnMoney}) \\
\textbf{means}(\mathsf{Withdraw}) &= \mathit{Cancel}(\mathsf{Pay}, \mathsf{Item}) \\
\textbf{means}(\mathsf{Reject}) &= \mathit{Release}(\mathsf{Pay}, \mathsf{Item})
\end{aligned}$$

And for the rest of $v \in V$, $\textbf{means}(v) = \mathit{None}$.

Let us remark two aspects of the proposed commitment specification language. First, it allows for embedded commitments. For example, the the definition of Offer uses Item, which creates a commitment itself (to return the money if the agent who paid complains). Second, operations over commitments do not have a debtor and a creditor. Instead, this information is implicit and depends on who utters the word and to whom, as we will see below. Intuitively, if a_1 sends Offer to a_2, the commitment will be created with a_2 as creditor and a_1 as debtor.

5.3.2 Semantics

The semantics of the specification language describes how agents can operate with commitments by sending messages to each other. Our ultimate goal is to define when an interaction between two agents *complies* with a specification, that is, when the agents have satisfied all the commitments they made. Before presenting the operational semantics, let us introduce the notion of messages and interactions.

Definition 5.3. Let $a, a' \in \mathcal{A}$ $(a \neq a')$ and $v \in \mathcal{V}$. A *message* is a tuple $m = \langle a, a', v \rangle$, where a and a' are the sender and receiver respectively.[1] An *interaction* is a sequence of messages $I = [m_1, \ldots, m_n]$. ■

[1]Since we only consider interactions between two agents, it would be enough to have one agent in the definition, but we keep two for clarity and compatibility reasons.

The operational semantics uses the notion of *commitment*. This notion is not explicit in the language, but only used to define its semantics.

Definition 5.4. A *commitment* is a tuple in the set $C = V \times V \times A \times A$. The agent IDs represents the debtor and the creditor respectively, and we require them to be different. The two words are the antecedent and the consequence of the commitment. ■

We will define the semantics of the language by describing the state of each possible commitment for an interaction I and specification **means**. A commitment is *inactive* when the debtor has not created it, *active* when the debtor created it, but its antecedent has not been uttered and *detached* when the debtor created it and its antecedent was uttered by the creditor.

Definition 5.5. Let **means** be a commitment specification over V. A *state function* is the function $\sigma : C \times I \to \{active, inactive, detached\}$ that is defined as follows. Let $u, v \in V$, and c be a commitment (u, v, a_1, a_2). The case when a_2 is the debtor is analogous.

$$\sigma(c, []) = inactive$$

$$\sigma(c, I'.\langle a_1, a_2, w\rangle) = \begin{cases} active & \text{if } \sigma(c, I') = inactive \text{ and} \\ & means(w) = Create(u, v) \\ inactive & \text{if } \sigma(c, I') = detached \text{ and} \\ & means(w) = Cancel(u, v) \text{ or } w = v \\ \sigma(c, I') & \text{otherwise} \end{cases}$$

$$\sigma(c, I'.\langle a_2, a_1, w\rangle) = \begin{cases} inactive & \text{if } \sigma(c, I') = detached \text{ and} \\ & means(w) = Release(u, v) \\ detached & \text{if } \sigma(c, I') = active \text{ and } w = u \\ \sigma(c, I') & \text{otherwise} \end{cases}$$

■

Our semantics is sequential: if a commitment is detached at some point, it has to be turned inactive, independently of what happened before. This allows for the specification of interactions in which the same commitment is made more than once. The state function σ always returns the last state of a commitment.

The states *active, detached* and *inactive* are enough to define the operational semantics of our specification language. We choose to use only these ones to maintain the definition short and clear. However, other notions that appear in the literature will be important for the inference process. Concretely, we will need to distinguish between different ways of turning a commitment from detached to inactive. When the debtor cancels it, we will say it is *canceled*, when the creditor releases it will be *released*, and when the debtor utters the consequent it will be *discharged*.

Finally, we define the notion of *compliance* of an interaction. Intuitively, an interaction complies with a specification if it has no detached commitments, that is, every commitment that was created and detached finished in one of the three possible ways.

Definition 5.6. An interaction I *complies* with a specification **means** if there are no detached commitments, that is, if $\sigma(c, I) = \mathit{inactive}$ or $\sigma(c, I) = \mathit{active}$ for every $c \in C$. ■

Commitments that are detached at the end of an interaction are normally called *violated*. As we will explain later, we assume all interactions agents see are compliant, so we will not need this distinction.

To illustrate these ideas, consider the buying example, given by the specification above. Consider the following interactions:

$$I_1 = [\langle a_1, a_2, \mathsf{Offer}\rangle, \langle a_2, a_1, \mathsf{Pay}\rangle]$$
$$I_2 = [\langle a_1, a_2, \mathsf{Offer}\rangle, \langle a_2, a_1, \mathsf{Pay}\rangle, \langle a_2, a_1, \mathsf{Withdraw}\rangle]$$
$$I_3 = [\langle a_1, a_2, \mathsf{Offer}\rangle, \langle a_2, a_1, \mathsf{Pay}\rangle, \langle a_1, a_2, \mathsf{Item}\rangle]$$

The first interaction does not comply with the specification, since the commitment $(\mathsf{Pay}, \mathsf{Offer}, a_1, a_2)$ is detached. I_2 and I_3 do comply, since the commitment is canceled and discharged, respectively. Note that another commitment is created in I_3 by saying *Item*, but it is never detached.

5.4 Inferring Social Semantics

The problem of inferring the commitment semantics of a community can be formulated as follows. Agents in a community interact following a commitment specification **means** over vocabulary V. Each interaction is between two agents a_1 and a_2, is finite, and complies with the specification. An external agent, called *student*, observes the complete interactions sequentially, one at a time. The student does not know neither the vocabulary V used by the agents nor the function **means**, but it assumes they are behaving according to some specification like the ones that we described in the previous section. Moreover,

the student does share with the rest of the agents the general operational semantics that we described in the previous section. That is, the student may not know that **means**(Offer) = *Create*(Pay, Item), but it knows that if it does, then $\sigma((\mathsf{Pay}, \mathsf{Item}, a_1, a_2), [\langle a_1, a_2, \mathsf{Offer}\rangle]) = \mathit{active}$. This, however, does not imply that the student uses the same words in the specification. It could perfectly well use any other word instead of *Create*; the important part is that they work in the same way.

In this section we present a technique that the student can use to infer the mappings in **means** only by observing interactions between agents that already know the specification.

A Technique to Infer Social Meaning

As in previous chapters, the student maintains a confidence value for each mapping between a word and a commitment operation. We represent these values as a function $\omega : V \times O_V \to \mathbb{R}^+$ which is updated with each observation of a new complete interaction, to reflect new evidence for possible mappings. We call V^ω the set of $v \in V$ such that $\omega(v, o)$ is defined for some $o \in O_V$. Similarly, for each $v \in V$, $O^\omega(v)$ are the commitment operations $o \in O_V$ for which $\omega(v, o)$ is defined.

The update works as follows. For each interaction, the student first computes the possible meanings for a word, and then it updates the value of those mappings in ω. Finding all possible meanings is challenging due to the existence of operations to finish commitments (*Cancel* and *Release*), combined with the fact that the student does not know the complete vocabulary a priori. As an example, consider again the *transaction* specification. Suppose the student observes the interaction $[\langle a_1, a_2, \mathsf{Offer}\rangle, \langle a_2, a_1, \mathsf{Pay}\rangle, \langle a_1, a_2, \mathsf{Withdraw}\rangle]$. Since it does not know a priori that *Item* exists, the student is unable to infer the correct semantics, and it may think that **means**(Offer) = *Create*(Pay, Withdraw). Our solution to this problem consists on inferring possible *Create* operations separately, by looking at the possible fulfilled commitments. With this information, the student can learn the rest of the operations.

From now on, we will consider a student that observes an interaction I with at least one message. For simplicity, we will refer to the word uttered in position i (with $1 \leq i \leq \mathit{len}(I)$) as v_i. From now on, to simplify the exposition, we will only work with commitments where a_1 is always the debtor. The case when a_2 is the debtor is analogous. First of all, the student computes possible *Create* meanings for each word in I. For an index i, these are the set of all commitment that can have been discharged after it, that we will call $\mathit{Disch}(I, i)$. From now on, we will refer with $I^{i:j}$ to the subsequence of I that starts at index i and finishes at index j.

Definition 5.7. Let I be an interaction and i an index such that $1 \leq i \leq \mathit{len}(I)$

and $I(i) = \langle a_1, a_2, v_i \rangle$ (everything is analogous for a_2). The set $Disch(I, i)$ contains all the commitments that may have been created by v_i and later discharged. Concretely, a commitment (v_j, v_h, a_1, a_2) is in $Disch(I, i)$ if there exist indexes j and h after i ($i < j < h < len(I)$ and $v_j \neq v_h \neq v_i$) such that $I(j) = \langle a_2, a_1, v_j \rangle$ and $I(h) = \langle a_1, a_2, v_h \rangle$. ■

As an example, consider the interaction $I = [\langle a_1, a_2, \mathsf{Offer} \rangle, \langle a_2, a_1, \mathsf{Pay} \rangle, \langle a_1, a_2, \mathsf{Receipt} \rangle, \langle a_1, a_2, \mathsf{Item} \rangle]$. The set of possible discharged commitments for index 1 is $Disch(I, 1) = \{(\mathsf{Pay}, \mathsf{Receipt}, a_1, a_2), (\mathsf{Pay}, \mathsf{Item}, a_1, a_2)\}$. The set of possible discharged commitments for index 2 is empty, because since a_2 spoke only once, it cannot have discharged any commitment.

To update possible *Create* meanings, the student rewards the mappings between each message v_i and the creation of all the commitments in $Disch(I, i)$. Let ρ_1 be a reward parameter. For all $1 \leq i < len(I)$, and for all $(v_j, v_h, a_1, a_2) \in Disch(I, i)$, it performs the following update:

$$\omega(v_i, Create(v_j, v_h)) := \begin{cases} \omega(v_i, Create(v_j, v_h)) + \rho_1 & \text{if } Create(v_j, v_h) \\ & \in O^{\omega}(v_i) \\ \rho_1 & \text{otherwise} \end{cases}$$

The process to update *Release* and *Cancel* operations is similar. First, the student generates possible meanings. Words that are candidates to mean *Release* are those that are uttered by the creditor of a commitment after the commitment was created and detached. *Cancel* candidates are the analogous ones that were uttered by the debtor of the commitment. Of course, the student ignores a priori which *Create* operations were actually uttered. For this reason we consider all *Create* meanings that are possible according to previous evidence. To compute possible *Cancel* and *Release* meanings, we first need to define the set $Det(I, i)$, that contains all possible detached commitments that were created by v_i.

Definition 5.8. Let I be an interaction and i an index such that $1 \leq i \leq len(I)$ and $I(i) = \langle a_1, a_2, v_i \rangle$. The set $Det(I, i)$ are all commitments that can have been created by v_i according to the student's ω and that are detached after i in I. Concretely, $(v_j, v, a_1, a_2) \in Det(I, i)$ if $Create(v_j, v) \in O^{\omega}(v_i)$, $I(j) = \langle a_2, a_1, v_j \rangle$ for some $i < j \leq len(I)$, and $\langle a_1, a_2, v \rangle \notin I^{j:len(I)}$. ■

As an example, consider the interaction $I = [\langle a_1, a_2, \mathsf{Offer} \rangle, \langle a_2, a_1, \mathsf{Pay} \rangle, \langle a_1, a_2, \mathsf{Item} \rangle]$, and suppose that $Create(\mathsf{Pay}, \mathsf{Receipt}) \in O^{\omega}(\mathsf{Offer})$. Then $(\mathsf{Pay}, \mathsf{Receipt}, a_1, a_2) \in Det(I, 1)$.

If a commitment is in $Det(I, i)$, and since all interactions comply with the specification, one of two possibilities is true. Either the *Create* meaning for that commitment is wrong for v_i, or it was canceled or released after it was detached. A commitment (v_j, v, a_1, a_2) in $Det(I, i)$ can be canceled by any word uttered

by a_1 after j, or released by those uttered by a_2 after j. These are the possible meanings to update.

Of course, since agents consider all possible *Create* meanings, some of them will be wrong. To take this into account, when updating the values for the possible *Cancel* and *Release* meanings in $Det(I, i)$ the student uses the confidence on the mapping between the corresponding *Create* operation and v_i. To this end, we use the values in a normalized version of ω. This function, called $\hat{\omega}$, is obtained by scaling the function ω in such a way that for each $v \in V^\omega$, $\sum_{o \in O^\omega} \hat{\omega}(v, o) = 1$, and for all $o \in O^\omega$, $0 \leq \hat{\omega}(v, o) \leq 1$.

For each $(v_j, v, a_1, a_2) \in Det(I, i)$ and for each $(a, v_h) \in I^{j:len(I)}$, let $o = Release(v_j, v)$ if $m = \langle a_2, a_1, v_h \rangle$, or $o = Cancel(v_j, v)$, if $m = \langle a_1, a_2, v_h \rangle$. With ρ_2 being another reward parameter, the update is as follows:

$$\omega(v_j, o) := \begin{cases} \omega(v_j, o) + \rho_2 \cdot \hat{\omega}(v, o) & \text{if } op \in O^\omega(v_h) \\ \rho_2 & \text{otherwise} \end{cases}$$

Information about possible *Cancel* and *Release* operations can also be helpful to update the *Create* meanings. There can be commitments in $Det(I, v_i)$ that cannot be canceled or released, because there are no utterances that can canceled them. This happens when the word that detaches the commitment is the last one in the interaction. For each commitment $(v_j, v, a_1, a_2) \in Det(I, i)$, if $j = len(I)$, the commitment cannot be created in the first place, and the student punishes the possible create with a parameter ρ_3:

$$\omega(v_i, Create(v_j, v)) := \omega(v_i, Create(v_j, v)) - \rho_3$$

5.5 Performance Analysis

We evaluated our inference technique using two different approaches. First, we analyzed how the student behaves when it uses the specifications it infers to interact with other agents. The second approach consists in measuring the correctness of the learned specifications with respect to the real ones.

Before describing the experiments, we need to establish which is the semantics that a student infers from the observations. This specification is obtained from the confidence distribution ω, and we will call it $\mathbf{means}^\omega$. It assigns to each word the possible meaning with maximal confidence value if there is enough evidence for it, that is, if its confidence is higher than the confidence on any other possible meaning by a difference larger than a given threshold. If there is not enough evidence, it assigns *None*:

Definition 5.9. Consider a student with confidence function ω and an evidence parameter ϵ_1. The specification $\mathbf{means}^\omega$ maps each $v \in V^\omega$ with a meaning in

$O^{\omega}(v)$ as follows:

$$\mathbf{means}^{\omega}(v) = \begin{cases} \underset{o \in O^{\omega}(v)}{\text{argmax}}\ \omega(v,o) & \text{if for all } o' \in O^{\omega}(v),\ \omega(v,o) - \epsilon_1 > \omega(v,o') \\ \mathit{None} & \text{otherwise} \end{cases}$$

■

We used randomly generated specifications and interactions in the experimentation. The only restriction we imposed to the specifications was that the *Cancel* and *Release* meanings were over commitments that could be created: to include $\mathit{Cancel}(v,w)$ (or $\mathit{Release}(v,w)$) as a meaning, we required that there existed a word with meaning $\mathit{Create}(v,w)$. For the interactions, we gave higher probability to the ones that had active and detached commitments. We performed experiments with vocabularies of 16 words.

5.5.1 Experiment 1

The first experiment analyses how often a student violates a commitment according to the original specification when it uses $\mathbf{means}^{\omega}$ to interact. We simulated interactions between agents that send messages to each other alternating turns. Each agent in the interaction follows its own specification, and it always complies with it. Concretely, agents always send messages that make it possible to comply with all the commitments in a predefined number of messages. We analyze how students that use a learned $\mathbf{means}^{\omega}$ interact with other agents.

In each experiment a student goes through a training phase in which it observes a given number of interactions that follow a specification **means**. We performed the experiment for training phases of different length. The observed interactions have random lengths of between 4 and 10 messages, to avoid choosing a length arbitrarily. After the training phase, we let the student interact with an agent 50 times, with each interaction having 6 messages. The student used $\mathbf{means}^{\omega}$ to interact, while the other agent used **means**. For each interaction, we checked if the student had complied with **means**. We performed the experiment for numbers of observed interactions between 50 and 350, repeating it 50 times for each value. Table 5.1 shows the average proportion of successes for each number of training observations. These results show that the student learns relatively fast specifications that allow it to interact with a marginal number of commitment violations.

5.5.2 Experiment 2

In the second experiment we compare the semantics in $\mathbf{means}^{\omega}$ with the original ones in **means**, using the F-score measure (see Definition 3.9).

50	100	150	200	250	300	350
55%	68.5%	76,4%	86%	88.5%	95%	96.5%

Table 5.1: Proportion of compliant interactions for different numbers of observed interactions

For each experiment, we generated a specification **means** and let a student observe, sequentially, interactions that complied with it. These interactions had variable length of between 4 and 10 messages. For each new interaction, we measured the F-score of $\mathbf{means}^{\omega}$ with respect to **means**. The experiment ended when the student found the correct meaning for all words (F-score($\mathbf{means}^{\omega}, \mathbf{means}$) = 1, and all the words in V are known), or when it had seen a limit of 1000 interactions. We used the values for parameters that had best performance in a preliminary test: the same value for ρ_1, ρ_2, and a much higher one for ρ_3.

Figure 5.1 shows the mean of the obtained F-scores as a function of the number of seen interactions for different types of specifications: (a) without *Cancel* or *Release* meanings, (b) with *Release* meanings, (c) with *Cancel* or *Release* meanings. In the figure it is clear how the inclusion of cancel meanings affects significantly the convergence to the correct specification. The results show that the commitment semantics that we presented, without any external restriction, is not possible to learn completely. This is because it is impossible to distinguish when an agent canceled a commitment from when it discharged it by uttering its consequent. For example, consider the following two specifications:

1. $\mathbf{means}(\mathsf{Item}) = \mathit{Create}(\mathsf{Complain}, \mathsf{ReturnMoney})$,
   $\mathbf{means}(\mathsf{ReturnMoney}) = \mathit{None}$

2. $\mathbf{means}(\mathsf{Item}) = \mathit{Create}(\mathsf{Complain}, \mathsf{Sorry})$, $\mathbf{means}(\mathsf{Sorry}) = \mathit{None}$,
   $\mathbf{means}(\mathsf{ReturnMoney}) = \mathit{Cancel}(\mathsf{Complain}, \mathsf{Sorry})$

Suppose the real semantics is the first one. Since agents only receive compliant interactions, $[\langle a_1, a_2, \mathsf{Item}\rangle, \langle a_2, a_1, \mathsf{Complain}\rangle]$ will always be followed, at some point, by $\langle a_1, a_2, \mathsf{ReturnMoney}\rangle$, but agents have no way of deciding if ReturnMoney discharges the commitment or cancels it. This situation is aggravated by the fact that the student never increases the confidence for *None* meanings. Although the semantics the student learns allows to interact correctly with others, they are actually using different meanings.

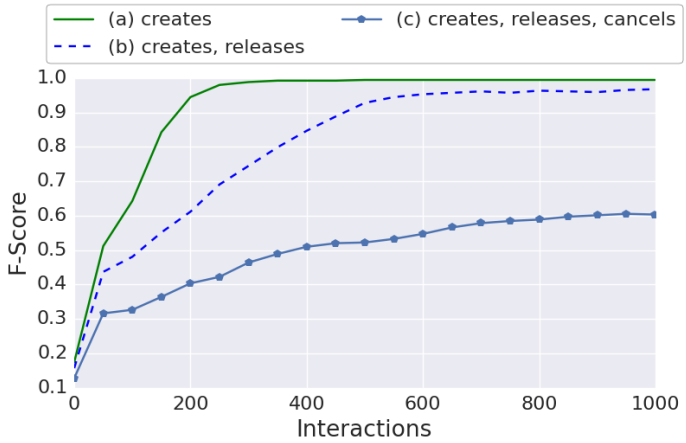


Figure 5.1: Convergence for different specification types

5.6 Semantic Extensions

The situation we just explained does not really affect the usability of the learning technique. For the language to make sense, there must be a difference between canceling a commitment and discharging it by uttering its consequent. Otherwise, there would be no need to have two operations. Indeed, the community working on commitments has proposed diverse ways of distinguishing between them. These differences are not in the operational semantics of the language, but are external factors, related to how the language is used. In the rest of this section we explain some of them and we investigate how our technique can take them into account, yielding better results in the second experiment.

5.6.1 Frequency

The simplest way of distinguishing between discharges and cancelations is by how frequently they occur. In (Singh 2012) it is argued that canceling a commitment should be an exceptional behavior, reserved for when for some reason agents cannot discharge it. For the notion of commitments to make sense, it is necessary that agents respect them most of the times. Considering a case in which the discharges are more common than cancelations is actually enough to improve the performance of our technique.

In a situation in which *Create* operations are more frequent than *Cancel* and *Release* ones, it makes sense to spend the first interactions trying to learn the first, making the division explicit. To implement this idea, we make the student update only *Create* possible meanings for some time, and then start including *Cancel* and *Release* operations. In the experimentation, we found that updating

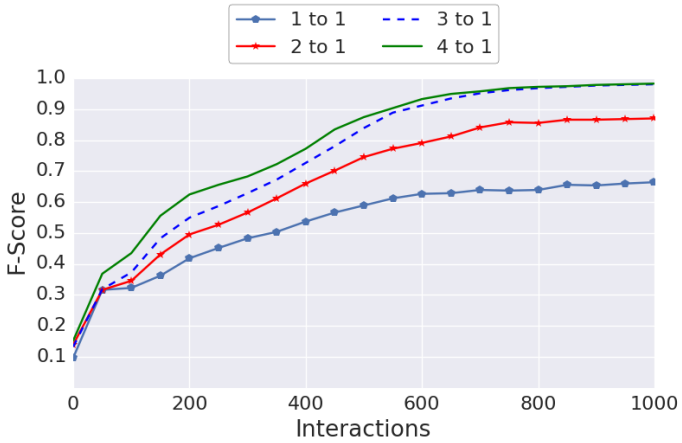


Figure 5.2: Frequency

only *Create* meanings for the first 10% of the total interactions yielded a good balance.

To test this idea, we built interactions in which agents are more likely to discharge than to cancel the commitments they made. Figure 5.2 shows the results for different ratios of cancelations to discharges. Considering discharges to be only twice as likely than cancelations already improves the performance notably. The modification updating only *Create* meanings for the first 10% interactions is actually better even for the 1 to 1 case, as it can be seen comparing this with Figure 5.1.

5.6.2 Observing Punishments

Another way of differentiating between discharges and cancelations takes into account the consequences for the agent that performs the action. This idea, developed for example in (Singh 2012), considers a difference between what agents *should* do and what agents *can* do. They should not cancel the commitments they make, but they can do it because it may be necessary in some cases. However, agents that cancel their commitments receive a punishment for their behavior. This mechanism allows to change the punishment in a flexible way, according to the commitment that has been canceled or to other factors. For example, the punishment for canceling a commitment to give an item if the creditor pays may be low if there is an emergency, but high otherwise.

Taking into account punishments for canceling commitments can help finding the correct social meanings. We consider a simple case in which agents receive a fixed punishment if they canceled any commitment in an interaction. The student can observe this punishment, as well as which agent was punished. If the

student observes an interaction and the information that agent a_1 was punished, it means that there is at least one commitment that a_1 created, a_2 detached, and a_1 canceled. Of course, the student does not know which commitment that is. However, if it observes that an agent was not punished, it knows that all its detached commitments were discharged or released, but not canceled. The student can therefore discard many possible *Cancel* meanings.

First of all, the student will only reward possible cancel meanings for those interactions when the agent was punished. For an interaction where a_i was not punished, the student subtracts a value ρ_4 from all possible *Cancel* meanings for all possible detached commitments. This is because we assume agents are punished when they cancel a commitment, not anytime they utter a word with a *Cancel* meaning. Concretely, consider indexes $1 \leq i < j < h \leq len(I)$, and let $I(h) = \langle a_1, a_2, v_h \rangle$ and $Cancel(u, v) \in O^{\omega}(v_h)$. If $(u, v, a_1, a_2) \in Det(I, i)$ and $I(j) = \langle a_2, a_1, v \rangle$, the student updates ω as follows:

$$\omega(v_h, Cancel(u, v)) := \omega(v_h, Cancel(u, v)) - \rho_4$$

Now the student is obtaining extra information about *Cancel* meanings, which can also be used to update *Create* meanings. This can be done by punishing all those *Create* meanings for which the operation is detached, and there is no possible *Cancel* or *Release* with high value later. Let $1 \leq i < j \leq len(I)$, and $I(i) = \langle a_1, a_2, v_i \rangle$, and suppose $Create(v_j, v) \in O^{\omega}(v_i)$ and $I(j) = \langle a_2, a_1, v_j \rangle$ but $\langle a_1, a_2, v \rangle \notin I^{j:len(I)}$ (it is not discharged). Let ϵ_2 be a parameter. If for all h such that $j < h \leq len(I)$, either $\hat{\omega}(v_h, Cancel(v_j, v)) \leq \epsilon_2$ if $I(h) = \langle a, v_h \rangle$ or $\hat{\omega}(v_h, Release(v_j, v)) \leq \epsilon_2$ if $I(h) = \langle a, v_h \rangle$ (or the meanings are not in $O^{\omega}(v_h)$), the original *Create* meaning is punished with a ρ_5 parameter:

$$\omega(v_i, Create(v_j, v)) := \omega(v_i, Create(v_j, v)) - \rho_5$$

Figure 5.3 shows the percentage of convergence for students that receive information about the cancel punishments. As it can be seen, the student is better at inferring commitment semantics when it can observe punishments, reaching high F-score. The technique still fails sometimes because agents fail to distinguish *None* from *Cancel* or *Release* meanings, but it no longer presents the problem that we described before.

5.6.3 Cancelation Policies

In the first papers about commitments (Singh 1999), the authors proposed to have an extra specification with regulations that are external to the commitment semantics. These are higher-order constraints that describe the conditions under which different operations over commitments can be performed. The conditions to cancel commitments, in particular, are specified by *cancelation policies*. In

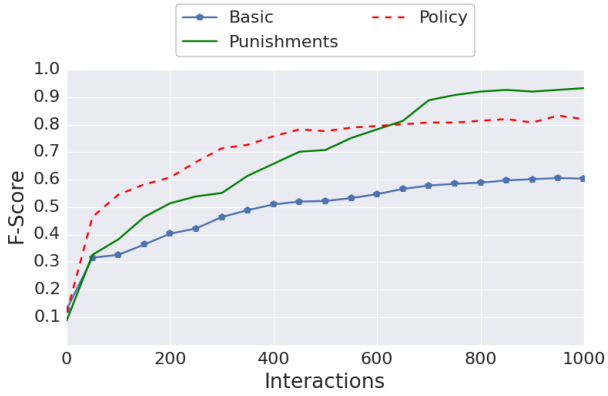


Figure 5.3: Punishments and policy

this section we study how this extra information can help in the process of inferring meaning.

Following our original idea that only utterances are observable, we define policies as sets of constraints over words, that establish what has to be said before being allowed to cancel a commitment.

Definition 5.10. Let $O^{cl}_{V\cup\{*\}}$ be all possible *Cancel* operations over the vocabulary V extended with a sign $*$. A *cancelation policy* is a relation $Pol : O^{cl}_{V\cup\{*\}} \times A \times V \times A$.■

If, for example, $(Cancel(u, v), a_1, w, a_2) \in Pol$, agent a_2 has to say w before a_1 can cancel a commitment (u, v, a_1, a_2). If the *Cancel* operation has $*$ as antecedent or consequent, the rule is valid for cancelations of commitments with any word in that position. As an example, a policy could say that a_1 can only cancel a commitment to give an item if there was an emergency, which has to be communicated by that same agent. This would be expressed as a social policy as follows:

$$(Cancel(*, Item), a_1, \mathsf{Emergency}, a_1) \in Pol$$

We can now define the notion of compliance of an interaction with a cancelation policy. The same is valid analogously if a_2 performs the cancelation.

Definition 5.11. Consider a vocabulary V and an interaction I, a specification **means**, and a cancelation policy Pol over $V \cup \{*\}$. An index $1 \leq i \leq len(I)$ such that $I(i) = \langle a_1, v_i\rangle$ and $\mathbf{means}(v_i) = Cancel(u, v)$ complies with Pol if, for $w \in V, a \in A$,

$$(Cancel(u, v), a_1, w, a) \in Pol \Rightarrow \langle a, w\rangle \in I^{1:i}$$

and the same is valid for $Cancel(*, v)$, $Cancel(u, *)$, and $Cancel(*, *)$. The interaction I complies with Pol if all its indexes comply with it. ■

Cancelation policies are common cultural knowledge that is universally shared, also by the student. Note that this implies also that the student needs to know (part of) the vocabulary a priori. These rules can be helpful to decide the meaning of words, since they can rule out impossible meanings. We assume that the interactions seen by the student always comply with the cancel policy, and that the student knows this.

The process to take cancelation policies into account is simple. When a student with a cancel policy Pol observes an interaction I, it checks each word for possible cancel meanings, and punishes those that would not comply with Pol. For each $1 \leq i \leq len(I)$ such that $I(i) = \langle a_1, a_2, v_i \rangle$, if $Cancel(u, v) \in O^{\omega}(v_i)$ and assigning that meaning would make the index i non-compliant, punish the mapping with a parameter ρ_6:

$$\omega(v_i, Cancel(u, v)) := \omega(v_i, Cancel(u, v)) - \rho_6$$

The effect of cancelation policies on the inference process depends on which kind of *Cancel* operations are regulated. If the policy has rules for *Cancel* operations with $*$ for both the antecedent and the consequent, the effect is similar than in the case with punishments, since the student knows that, in some situations, the agents cannot have canceled their commitments. Figure 5.3 shows the results for a student that shares a policy with the community. We used policies that had one rule for each *Create* meaning in the original specification, with $*$ as consequent. These rules are the most informative ones, since in the case without policies the student confuses canceling operations with the same antecedent. As we can see, agents reach higher F-score sooner than in the case with punishments. This may be because every interaction is affected by the policy, while interactions where agents cancel commitments give no extra information in the case of cancelation policies. However, they obtain a lower value in the end, which may be because the restrictions imply that some words are uttered less often than others, and the student does not have enough information to learn their meaning.

5.7 Conclusions

In this chapter we presented a method to infer the commitment semantics used by a community, that can be used by a student who observes compliant interactions between other agents. Importantly, our techniques use no external resource, and make no assumption about the student, except that it knows the operational semantics of the specification language. The actual language used

in the specification, however, can be different. This provides a very flexible method that can be used in open situations.

The problem that we solve in this chapter is different from the ones we tackled in previous ones. Concretely, in here we consider the inference of a semantics, instead of an alignment, and the agent that learns is external instead of one of the interlocutors. This setup makes the problem simpler and more complicated at the same time. It is simpler because, since all agents who interact know the semantics, there is no uncertainty on whether they have interpreted things correctly. It is more complicated because there is no pre-defined set of possible meanings, but any combination of commitment operations could be one. We think the methods we proposed here to update the possible meanings could be used for the problem of aligning vocabularies with commitment semantics, but this needs to be adequately investigated.

The semantics of commitments as we defined is not possible to learn completely by observing interactions. This is because it does not distinguish between the acts of canceling and of discharging a commitment. We explored different extensions to the semantics that help to differentiate between these two actions, and showed how they are useful to infer the correct semantics. Already observing agents that discharge their commitments more frequently than they cancel them results in much better results in the inference process.

In this chapter we only performed experiments with small vocabularies, that are useful to show how the techniques work, but are not realistic use cases. We performed exploratory experiments with larger vocabularies that show that, while the techniques seem to scale up in terms of finding good alignments, they become very slow for large vocabularies. This is because there are many possible meanings for each words. A possible way to take this into account is to use a pruning technique on the possible mappings. Another possibility is to consider pragmatic restrictions, such as not allowing agents to cancel commitments that have not been created.

Chapter 6

A Study with Empirical Data [1]

> "If you wish to make an apple pie from scratch, you must first invent the universe."
>
> Carl Sagan, *Cosmos*

The techniques that we proposed in the past three chapters were applied over protocols defined in a very formal and general way. Not making assumptions about the implementation techniques of the specification languages allowed us to provide general methods can be adapted to different concrete cases. However, it also implied that the techniques cannot be applied to real-world data directly, without first making some modifications. For this reason we chose to evaluate our methods only on artificially generated protocols, providing a report of how our methods perform abstractly, without dealing with the implementation details. This allows our evaluation to cover scenarios of different complexities, and avoids the bias that a particular dataset would inevitably introduce.

In this chapter we report the results of our efforts to apply the ideas that we discussed throughout this dissertation on an empirical dataset. Concretely, we use a set of procedural human-crafted protocols in Spanish and English from an online collaborative webpage, that we formalized into a machine-readable format with existent techniques. We consider a scenario where two agents interact following these protocols. As before, they share the procedural structure but speak different languages. We propose a translation inference technique that combines the ones in Chapters 3 and 4, with modifications to address the challenges of working with natural language labels. We performed two types of evaluation. First, we measured how well agents can interact with each other when using our aligning technique, and we show that they are better than when using a

[1] This chapter reports results from work performed jointly with Paolo Pareti, from the University of Southampton.

dictionary. Second, we analyze, in different ways, the semantic correctness of the obtained mappings.

6.1 Introduction

Having machines understand instructions formulated in natural language is not only a goal of artificial intelligence, but already a reality. Almost every cellphone can now understand simple instructions to message contacts, play songs, or program alarms. While not long ago search engines had their specific query languages to specify searches, nowadays users mostly just ask questions, to which engines increasingly give meaningful answers.

Research on how to automatize the understanding of natural language instructions shows that we are also on the way of developing machines that can understand and perform complex, sequential tasks. These can be complicated to read, since different steps can have dependencies between them and with the external world. For example, if an instruction indicates to *beat the egg whites with the sugar*, it may be evident for humans that both ingredients have to be first put into a bowl. Machines, instead, need to infer this. Some examples of approaches that provide a way of automatically formalizing step-by-step instructions described in natural language into machine-understandable data are in the work by Addis and Borrajo (2010), Kiddon et al. (2015), Tenorth et al. (2010), Schumacher et al. (2012) and Malmaud et al. (2014). In this chapter we will focus on the work by Pareti et al. (2014), which provides a formalization that allows agents to automatically understand and execute instructions, provided they have the necessary abilities (see also the work in (Pareti et al. 2016) and (Pareti 2016) for more information on these techniques, as well as (Pareti 2017c) for a comprehensive exposition of the methods).

Collaboration is a key part of being able to perform complex sequential tasks. Humans following a recipe are not expected to perform every step on their own. If the recipe calls for, say, apples, it is understood that the human following it can rely on somebody else to grow and harvest the apples. In the same way, it is likely that one machine will not be able to perform all the steps on its own, so it is important that it is able to identify when it needs to collaborate and with whom.

To introduce an example that we will use throughout the chapter, consider an agent trying to make a cup of tea. Suppose it needs to follow three steps: 1. get a tea bag; 2. get hot water; 3. put tea bag in water. Collaboration is strictly necessary when the agent is unable to perform some of the steps in the task. For example, if the agent has no authorization to access any heating device, it will need help to perform the second step. Even when agents can perform the complete protocol, collaboration may be more efficient than working alone,

since tasks can be executed in parallel.

To allow for collaboration, as we have discussed in previous chapters, it is essential to ensure meaningful communication. This is particularly difficult when performing particular tasks that require domain-specific terminology. Names for specific tools or activities are notably diverse between communities of speakers, even within the same language. For example, a British agent trying to make tea would heat water on a *cooker*, while an American one would use a *stove*. Additionally, the same word can be used for different things: asking for a *screwdriver* in a hardware store can have different results than in a pub. In the former case a tool to manipulate screws is being requested, while in the pub a screwdriver is understood to be a cocktail.

Most of the techniques to formalize how-to instructions are developed for mono-language tasks, without considering the problem of collaboration between agents that speak different languages. In this chapter we explore how such collaboration can be achieved without using any external resources, applying the techniques that we developed in the previous chapters of this dissertation. To this end, we combine ideas from Chapters 3 and 4. The real-world protocols that we use in this work are expressed as automata, similarly to those in Chapter 3. However, they are much less informative, and a single protocol is not enough to define a translation. For this reason we considered a cross-situational approach like the ones in Chapter 4. We adapted our techniques to consider labels written in natural language. This raised three main challenges:

1. Messages are not single words, but sentences that were usually not repeated, so keeping a confidence for mappings between sentences was not very useful.

2. Words are not uniformly uttered in natural language, and some are much more common than others.

3. It is not always possible to directly map words between multiple languages. For example, some concepts are described by a single word in a language and by a multi-word expression in another one. For example, the Spanish translation for *applesauce* is *compota de manzanas*.

In this chapter we address the first and second issue, leaving the third one for future work. The contribution of this chapter is twofold. First, we test the interaction-based vocabulary alignment techniques that we proposed previously with a real-world instructional dataset. In this way we uncover potential challenges that may arise in the process, and we provide solutions to solve these challenges. At the same time, we provide a novel automatic tool to allow artificial agents that speak different languages to collaborate when following human-crafted protocols. It should be noted that our technique does not rely

on external linguistic resources, neither does it make strong assumptions about the languages that are being mapped. This yields a general method that can be used even by agents that do not know the language used by their interlocutor.

Methodology and Structure of the Chapter

We used human-crafted protocols obtained from the WikiHow website, a collaborative platform where people can submit their instructions to perform different tasks. According to WikiHow, they have more than 190,000 how-to articles and 1.7 millions registered users (information available in `https://www.wikihow.com/wikiHow:Statistics`, accessed on 21-02-2018). One of the premises of the platform is that it should be multilingual, to reach as many people as possible. Currently, there are instructions in 17 languages, and they are working to increase this number. In many cases, a how-to page is translated to different languages, which is particularly useful for our purposes. We extracted these protocols using existing methods (Pareti et al. 2014). Concretely, we obtained protocols to perform the same tasks in English and Spanish, restricting the tasks to the domain of cooking recipes.

We evaluated the adapted techniques that we propose with simulations that apply them on the extracted protocols using two measures. First, we studied how well agents perform the tasks when they use the translations they inferred. Importantly, this evaluation considers the semantic correctness of the execution, while the alignment techniques that we propose only use structural properties of the protocols. We use the Oxford Dictionary (Stevenson 2010) as a benchmark, and we show that agents that use our technique have better performance. Second, we analyzed the quality of the translations that were obtained. To this aim we compared them to the dictionary, and also performed two types of human evaluation.

We provide details of the data extraction in Section 6.5. Before turning to that, Section 6.2 describes the protocols in more general terms. When possible, we provide examples of interoperability between two protocols in English, to make them understandable for all readers. In Section 6.3 we describe the setting of the collaboration. Section 6.4 presents the adapted interaction-based alignment technique, showing how to deal with the challenges that arise from using natural language. Section 6.6 describes the evaluation of the techniques when used by agents following protocols in our corpus. We measured the percentage of successful interactions achieved by the agents as well as the precision of those translations.

6.2 A Model of Protocols

Intuitively, a procedural protocol is a set of instructions that must be performed to obtain some goal, together with a *dependency relation* that specifies an order between these instructions.

Definition 6.1. (Protocols) Let V be a vocabulary and V^* be the set of sentences in V, that is, the set of sequences of elements in V, that we will call *tasks*. A *protocol* over V is a tuple $\langle T, \prec \rangle$ where $T \subseteq V^*$ and $\prec$ is a strict partial order $\prec$: $T \times T$ that represents the *dependencies* between tasks. More specifically, $\mathtt{t} \prec \mathtt{t}'$ means that task $\mathtt{t}$ must always precede task $\mathtt{t}'$. ■

Note that these protocols are similar to the ones used in Chapter 3, but unlike those, they do not specify a unique order of tasks, since $\prec$ is partial. As an example, consider the following protocol $Tea = \langle T, \prec \rangle$ that specifies how to achieve the goal of making tea. We rename the tasks for simplicity.

- $T = \{$ "Get a tea bag" (`tea bag`), "Get hot water" (`water`), "Put tea bag and water in the cup" (`cup`) $\}$
- $\mathtt{water} \prec \mathtt{cup}, \quad \mathtt{tea\ bag} \prec \mathtt{cup}$

In this work we make the assumption that tasks are performed sequentially, forming *executions*.

Definition 6.2. (Executions) Let $\mathfrak{P} = \langle T, \prec \rangle$ be a protocol. An *execution* of $\mathfrak{P}$, noted E, is a sequence of tasks in T. A *successful* execution of $\mathfrak{P}$ is a sequence of tasks E that satisfies two criteria:

- The sequence E contains all and only the tasks in T.
- The order of tasks in E does not violate the dependency relation $\prec$. That is, for all $\mathtt{t}, \mathtt{t}' \in T$, if $\mathtt{t} \prec \mathtt{t}'$ and there exists a sequence E' such that $E = E' \,.\, \mathtt{t}'$, then $\mathtt{t} \in E'$. ■

Executions [`tea bag`, `water`, `cup`] and [`water`, `tea bag`, `cup`] are examples of successful executions of this protocol. In the first one, the tea bag is obtained before getting water, while in the second one, the tea bag is obtained just before adding it to the cup. An example of an unsuccessful execution is: [`tea bag`, `cup`, `water`], which involves putting a tea bag in a cup of water before getting the water, violating the dependency between `water` and `cup`.

6.3 Performing Tasks Collaboratively

As before, we work with a message-based collaboration scenario. Consider two agents that want to work together to achieve some goal. Agents perform tasks individually and have their own local representation of the execution. When one of them completes an action, it communicates this to its partner by sending a message with the label of the completed task. When an agent receives a message from its partner, it adds the corresponding task to its local execution. In this way, a procedural protocol is turned into an interaction protocol, and executions into interactions.

Of course, in some cases sending a message to communicate the performed task can be unnecessary, since agents may be able to perceive the changes and infer the performed action. For example, an agent can see that its partner performed "Get hot water" if there is a cup full of hot water. This is not always the case, and whether agents can or cannot identify completed tasks automatically depends on their perceptual abilities. For example, an agent may not realise that its partner got hot water if it does not have a temperature sensor. The collaborating dynamics that we propose are independent of such abilities: agents only need to understand the messages their partners send. Joint perception would be an addition to this basic assumption.

Collaboration is simple when agents share the same protocol. When an agent receives a message, it can immediately know which task was performed by its partner. Here we focus on the case in which agents share the dependency structure of the protocol, but not the vocabulary of tasks. As in previous chapters, the notion of *compatibility* between protocols characterizes this situation.

Definition 6.3. (Translations and Compatibility)

Let V and V' be two vocabularies. A *translation* is a partial function $\tau : V^* \to V'^*$. We will apply translations also to subsets of V^*.

Let T, T' two sets of tasks over V and V' respectively, and $\prec$ and $\prec'$ be order relations over T and T'. Let $\mathfrak{P} = \langle T, \prec \rangle$ and $\mathfrak{P}' = \langle T', \prec' \rangle$ be two protocols. We say $\mathfrak{P}$ and $\mathfrak{P}'$ are *compatible* if there exists a bijective translation τ such that, for each execution E that is successful for $\mathfrak{P}$, the execution that results from translating each task with τ, called $\tau(E)$, is successful in $\mathfrak{P}'$. When we know the translation τ, we will say that the protocols are *compatible under* τ. ■

Note that we defined the compatibility already with bijective translations. Intuitively, the notion of compatibility implies that the two protocols have the same number of tasks and that the structure of their dependencies is the same. To illustrate these ideas, consider another protocol $Tea' = \langle T', \prec' \rangle$ with a different procedure to make tea:

- T' = "Get a tea bag" (`tea bag'`),"Get hot water" (`water'`), "Put the tea bag in a cup" (`put'`), "Pour the water in the cup" (`pour'`) }
- `tea bag'` $\prec'$ `water'`, `water'` $\prec'$ `pour'`, `put'` $\prec'$ `pour'`

The protocol *Tea*$'$ is not compatible with *Tea*, since all its successful executions have four tasks instead of three. Instead, consider a third protocol *Tea*$''$:

- T'' = { "Get tea bag" (`tea bag''`), "Microwave cup of water for 3 minutes" (`water''`), "Add the tea bag to the cup" (`add''`) }
- `tea bag''` $\prec''$ `add''`, `water''` $\prec''$ `add''`

Tea$''$ is clearly compatible with *Tea*, under a translation τ such that τ(`tea bag`) = `tea bag''`, τ(`water`) = `water''`, τ(`cup`) = `add''`. Notice that the mapped labels are not a literal translation: "Microwave cup of water for 3 minutes" is not the same as "Get hot water". Moreover, *Tea*$''$ is also compatible with *Tea* under another translation τ', in which τ'(`tea bag`) = `water''`, τ'(`water`) = `tea bag''`, and τ'(`cup`) = `add''`.

This last point is important. Two protocols being compatible under τ does not imply that every label t is semantically equivalent to $\tau(t)$. The notion of compatibility is structural and not semantic. As we will explain later, we work under the implicit assumption that there exists one τ under which protocols are compatible which is also meaningful semantically. However, the translation technique is only defined in structural terms, without any semantic resource.

6.3.1 Collaboration Dynamics

From now on, consider agents a_1 and a_2 that can collaborate to achieve different goals, but use two different vocabularies V_1 and V_2 respectively. For each of these goals, let T_1 and T_2 be sets of tasks over V_1 and V_2 and $\prec_1$ and $\prec_2$ partial orders over the sets of tasks. Agents a_1 and a_2 have protocols $\mathfrak{P}_1 = \langle T_1, \prec_1 \rangle$ and $\mathfrak{P}_2 = \langle T_2, \prec_2 \rangle$ respectively. We assume both protocols are compatible, in particular under a translation τ. We will use these names to refer to a generic pair of protocols that we will use thoughout the paper.

We suppose a_1 and a_2 need to collaborate, since they are unable to perform all the tasks by themselves. We call $K_1 \subseteq T_1$ and $K_2 \subseteq T_2$ the set of tasks that a_1 and a_2 can perform respectively. The only restriction to be able to work together is that, by acting jointly, they can perform the complete protocols. That is, $T_1 \subseteq K_1 \cup \tau(K_2)$ for some τ under which $\mathfrak{P}_1$ and $\mathfrak{P}_2$ are compatible.[2]

[2]To simplify the implementation we require that K_1 and $\tau(K_2)$ are disjoint, but this plays no role in the learning method.

Agents send messages in their own languages, and interpret the ones received by their interlocutor. As before (see Section 4.3), we call *local* the executions maintained by each agent in their own vocabulary, and *global* the sequence of real performed tasks, in both vocabularies. We now describe how tasks are chosen and how messages are interpreted.

After a local execution E_1, agent a_1 can perform any *possible* task $\mathtt{t} \in T_1$, as it is defined in what follows. The definition is analogous for agent a_2.

Definition 6.4. The set of *possible tasks* for agent a_1 after E_1, denoted $Poss_1$, are all tasks $\mathtt{t} \in T_1$ such that

1. $\mathtt{t} \in K_1$,
2. $\mathtt{t} \notin E_1$, and
3. for all $\mathtt{t}'$ such that $\mathtt{t'} \prec \mathtt{t}$, $\mathtt{t'} \in E_1$ ■

Note that, with the second condition, we do not allow for executions with the same task. This is to ensure that agents progress and do not repeat always the same step. The definition of successful executions allows for repeating tasks, but for all protocols there exist an execution without duplicates that is successful.

To interpret the messages received in a foreign language, agents use a local translation, that we will call τ_1 and τ_2 for agent a_1 and a_2 respectively. These functions translate foreign sentences to local ones, that is, $\tau_1 : V_2^* \nrightarrow V_1^*$ and $\tau_2 : V_2^* \nrightarrow V_2^*$ When it receives a foreign message m, agent a_1 finds $\tau_1(m)$ and chooses it as an interpretation, adding it to the execution. Agents finish the interaction when either their local executions are successful, or none of them has possible messages, that is, $Poss_1 = Poss_2 = \varnothing$.

Two points should be noted. First, there is no guarantee that the protocols are compatible under local translations. When they are not, agents will finish the interaction before obtaining successful local executions, since they misinterpreted some task. Second, even if agents finish with successful local executions, the joint execution may not be semantically correct. For example, if a_1 uses *Tea* and a_2 uses *Tea*$''$, they could perform the execution [`tea bag`, `tea bag''`, `cup`]. This translates to successful interactions under the previously described τ', but it results in agents obtaining two tea bags and no water, which clearly violates the semantic interpretation of the task. In the next section we present a technique to learn a local translation that optimizes successful local execution, making agents obtain successful interactions more often. The notion of semantically correct execution is not used here; moreover, we do not assume there is a way of deciding if an execution is semantically correct or not in the collaboration scenario. We will later show experimentally that the translations which are computed by our technique to optimize successful local executions also lead to semantically correct executions.

6.4 Translation Learning Technique

To optimize local successes, agents need to find translations under which their protocols are compatible. We now explore whether these translations can be inferred only from the experience of interacting. In previous chapters, we did this by making each agent maintain and update an alignment, which was a confidence distribution over possible mappings between foreign and local words.

A confidence distribution for mappings between messages is useful when messages are indivisible units that are frequently repeated, or when agents interact with a reduced set of protocols. This approach might not be the best choice in our setting, however, given that tasks are sets of words and that there is no limit on the number of protocols that can be used. For example, consider an agent that learns, from repeated interactions, that the sentence "Put the tea bag in the water" maps to the Spanish sentence "Poner la bolsita de té en el agua". Even being correct, this translation might not be useful for future interactions. While the words found in these sentences (such as *water* or *tea*) are likely to be encountered again, the agent will probably never receive the exact same message in a different protocol.

The method we propose computes two confidence distributions: one between words and one between sentences. We present the technique to learn τ_1 from the point of view of agent a_1. The technique applied by agent a_2 is analogous. Agent a_1 has two confidence distributions. The first one, $\omega : V_1 \times V_2 \to \mathbb{N}$, is an alignment between words similar to the ones in Definition 3.8, that is, a partial function that assigns confidences to mappings between foreign and local words. This distribution is stored by the agent, and it is updated when new evidence is obtained. The second partial function, $\delta : {V_1}^* \times {V_2}^* \to \mathbb{N}$, maps foreign and local sentences, and it is computed using ω.

When it receives a foreign message $\mathtt{t_2}$, agent a_1 performs two actions. First, it computes its expected messages in $Poss_1$ and updates ω for the words in $\mathtt{t_2}$ with this information. Second, it computes δ to choose a local interpretation for the foreign sentence. The two confidence distributions interact with each other. On one side, the similarity of two sentences is computed using the values of the mappings between their words. On the other side, the update of the word-level mapping confidences takes into account the whole sentences in which words appear.

This approach makes two assumptions. First, to compute sentence similarities from word similarities it is necessary to assume that the meaning of a sentence is related with the meanings of the words that appear in it. Second, using the full sentences to determine word similarities is only useful if similar words tend to appear surrounded by words that are also similar. These assumptions are close to the hypothesis of *distributional semantics* (Turney and Pantel 2010, Baroni and Lenci 2010) known as *principle of compositionality* and *distri-*

butional hypothesis respectively. Indeed, the spirit of the methods we propose is similar, but distributional semantics techniques assume the existence of a large corpora to extract information about word similarity. Our approach, instead, works with much sparser data, but harnesses shared structure, which in this case is a protocol.

Finally, it is important to note that our approach, in its current form, still relies on mapping words between each other. This works well for pairs languages that have a similar conceptualization of words, that is, in which the mapping of words to concepts is more or less similar. However, this can be problematic when this does not happen. For example, mapping German instructions to Spanish ones may be more difficult with our technique, since in German words can be composed, forming one word that describes a complex concept. Such a word would not have any equivalent in Spanish. In Section 6.6.3 we discuss this aspect. Now, we explain how confidence distributions are updated.

6.4.1 Choosing a Mapping

Still from the point of view of a_1, finding an interpretation for a received message $\mathtt{t_1}$ involves two steps:

1. Computing δ from ω.
2. Computing τ_1 from δ.

The second step is straightforward. The local interpretation of message $\mathtt{t_2}$ is chosen randomly between the possible local tasks that map with $\mathtt{t_2}$ with maximal confidence:

$$\tau_1(\mathtt{t_2}) = \mathit{random}(\underset{\mathtt{t_1} \in Poss_1}{\operatorname{argmax}}(\delta(\mathtt{t_1}, \mathtt{t_2}))$$

Obtaining the values of the sentence mappings from ω requires more work. The underlying intuition is that the confidence for the mapping between two sentences needs to be computed from the confidences of the mappings between their words. The problem to be solved is how to combine the words, taking into account that word ordering is different for each language. For example, adjectives precede nouns in English, while in Spanish it is generally the other way round.

There exist multiple approaches to computing similarity between sentences (Li et al. 2006, Agirre et al. 2016, Mihalcea et al. 2006). These techniques require either external resources such as semantic databases, large corpora of data, or information about the grammar of particular languages. In our work we do not assume these resources to be available. We present an approach to combine the information about individual word mappings that, in spite of its simplicity, is

effective in dealing with simple imperative sentences such as the ones commonly found in how-to instructions. We solve the problem of not knowing which words should be mapped between each other by considering all possible mappings, and choosing the one with highest confidence. Concretely, we consider all possible combinations of mappings.[3]

Let $\mathtt{t_1} \in {V_1}^*$ and $\mathtt{t_2} \in {V_2}^*$. That is, for some $m, n \in \mathbb{N}^+$, $\mathtt{t_1}$ is a sequence of m words in V_1 and $\mathtt{t_1}$ is a sequence of n words in V_2. Suppose, in this case, that $m \leq n$ (otherwise everything is analogous). The value $\delta(\mathtt{t_1}, \mathtt{t_2})$ is computed as follows. Consider all the permutations of length m of the sentence $\mathtt{t_2}$, that we will call $Perm_m(\mathtt{t_2})$. For each $c \in Perm_m(\mathtt{t_2})$, we compute its *partial confidence degree* δ_p with $\mathtt{t_1}$ by considering the mappings between words in the same indexes:

$$\delta_p(\mathtt{t_1}, c) = \sum_{1 \leq i \leq m} \omega(\mathtt{t_1}(i), c(i))$$

Then, the confidence between the original words is the maximal partial value:

$$\delta(\mathtt{t_1}, \mathtt{t_2}) = \max_{c \in Perm_m(\mathtt{t_2})} \delta_p(\mathtt{t_1}, c)$$

This way of computing δ does not take into account the difference in length between the sentences. Although this should not be considered too strongly for the reasons we have already mentioned, this information can be helpful, particularly in the first interactions when there are too many options. To take this into account, we subtract to $\delta(\mathtt{t_1}, \mathtt{t_2})$ a value that is computed in relation to the difference in their length. Given a constant parameter ρ:

$$\delta(\mathtt{t_1}, \mathtt{t_2}) := \delta(\mathtt{t_1}, \mathtt{t_2}) - \rho \mid n - m \mid$$

Taking Word Frequencies into Account

The technique we proposed still does not take into account an inherent property of natural languages: some words are more frequent than others. Words that appear more frequently will be updated more often, and therefore their mapping with any other word will have a higher value. To take this into account we propose to incorporate information about word frequency to the computation of δ. Coherently with our assumption that agents have no previous information, we propose to track the frequency of foreign words dynamically. Concretely, agent a_1 maintains a partial frequency function $Freq : V_1 \cup V_2 \nrightarrow \mathbb{N}$. The function is partial because it only has values for those words that have already been

[3]This is computationally expensive, but it is also the most general possibility. A cheaper approach would need information about the specific languages to decide which combinations are worth considering.

used at least once. For local words $v \in V_1$, $Freq(v)$ is updated each time the agent starts using a new protocol, counting how many times they appear. For foreign words, it is updated each time the agent receives a new sentence. Notice that this takes into account how many times each protocol is used. If this is not desirable (for example, because protocols are used many times in a row, and then never again), agents can alternatively update frequencies only the first time a protocol is used. When computing the partial mapping degree between two sentences, the value of each mapping is divided by these frequencies.

$$\delta_p(\mathtt{t_1}, c) = \frac{\sum\limits_{0 \leq i \leq m} \omega(\mathtt{t_1}(i), c(i))}{Freq(\mathtt{t_1}(i)) + Freq(c(i))}$$

6.4.2 Updating ω

The technique to choose interpretations relies on ω, which represents the confidence of mappings between individual words. We now discuss how these values are updated from the experience of interacting.

When agent a_1 receives $\mathtt{t_2}$, it first updates the mappings between the words in $\mathtt{t_2}$ and the words in possible messages. Let r be a constant parameter and $v_2 \in \mathtt{t_2}$. We assume the agent initializes $\omega(v_1, v_2) = 0$ for all $v_1 \in V_1$ the first time it receives v_2. Then, for all $\mathtt{t_1} \in Poss_1$ and $v_1, v_2 \in \mathtt{t_1}$:

$$\omega(v_1, v_2) := \omega(v_1, v_2) + r$$

If v_1 does not belong to a possible message, the value of its mappings remains the same. Using only this simple approach, agents can be overlooking useful information that would be easy to take into account. For example, about the other words that appear in the sentence. As an example, suppose an agent following a protocol in Spanish receives the sentence t_{en} = *cup of barley*. The agent may not know the word *barley*, but if it has performed some other protocols before, it is probably familiar with *cup* and knows that it maps to *taza*. If the agent has a local task t_{es} = *taza de cebada*, it may infer that t_{es} maps with t_{en} only using *cup*, and in this way learn that *barley* maps with *cebada*.

Conveniently, the information about the other words in a sentence is in the already computed confidence values for mappings between sentences. Agents only need to also use these values to update the confidence degree.

Again, suppose a_1 receives $\mathtt{t_2}$. For all $\mathtt{t_1} \in Poss_1$, consider all $v_2 \in \mathtt{t_2}$ and all $v_1 \in \mathtt{t_1}$. Assuming again that $\mathtt{t_1}$ is shorter than $\mathtt{t_2}$, Let $Perm_{v_1,v_2}$ be all the permutations c of $\mathtt{t_2}$ of the same length as $\mathtt{t_1}$ such that the index of v_2 in c is the same as the index of v_1 in $\mathtt{t_1}$. Since the v_2 only maps with one word, the agent updates ω as follows:

$$\omega(v_1, v_2) := \omega(v_1, v_2) + \max_{c \in Perm_{v_1,v_2}} \delta_p(\mathtt{t_1}, c)$$

In our technique, agents perform first the simple update and then, once the mappings are computed they add these values.

6.5 Data Acquisition

In order to evaluate our techniques in a concrete scenario we tested them against real-world human-crafted protocols. We obtained these protocols from the Human Instructions Dataset, a publicly available dataset of formalized instructions (Pareti et al. 2014). This dataset contains over 200,000 sets of instructions extracted from wikiHow (`http://www.wikihow.com/`) and other similar websites. A multilingual version of this dataset, (Pareti 2017b) which only contains instructions from the wikiHow website, is available in 16 different languages. In this work, we have focused on the English and Spanish subsets of this dataset which contain, respectively, 133,842 and 120,507 sets of instructions.

In this dataset, sets of instructions are formalized as graphs using the PROHOW instructional model (Pareti et al. 2014) and are serialized as RDF triples (Consortium et al. 2014). The PROHOW model represents instructions using the concepts of *steps*, *methods* and *requirements*. Intuitively, steps decompose a set of instructions into a set of simpler tasks; methods provide information about different ways to decompose a a set of instruction into steps; and requirements provide a notion of dependency between tasks. The PROHOW model specifies a logical interpretation of these terms with respect to their execution. For example, this model allows to infer that a set of instructions is complete if all of its steps have been accomplished, or that a particular task is not ready to be executed if some of its requirements have not been provided.

The PROHOW model can be seen as a generalization of the protocol model defined in Section 6.2. For this reason, we developed a preprocessing pipeline to convert PROHOW instructions into protocols suitable for our experiment. This pipeline involves three phases: (1) instruction selection, (2) conversion of instructions into protocols and (3) protocol selection.

6.5.1 Instruction Selection

Not all sets of instructions can be translated into protocols, as some features of PROHOW are not expressible in our protocol model. For this reason, we excluded from our experiment sets of instructions which contained multiple methods or alternative sets of requirements. The software used in this preprocessing pipeline, along with details of its exact configuration, is available on GitHub (Pareti 2017a).

We focused our experiment on a particular domain of human instructions, namely cooking recipes, to obtain protocols with domain-specific language and words that are repeated among different sets of instructions. This was done by considering only instructions that belong to 8 wikiHow categories related to cooking. Since we are interested in protocols in English and Spanish, we excluded instructions that are not available in one of these languages.

After this selection process, the remaining sets of instructions only contained a linear sequence of steps and a set of necessary requirements, which are classified as either *tools* (non-consumable requirements) or ingredients (consumable requirements). Since we were interested in obtaining compatible pairs of protocols, we excluded pairs of instructions which do not contain the same number of steps, tools and ingredients. Sets of instructions which differ in the number of such elements, in fact, are modeled in different ways and they cannot be immediately mapped into each other. The selection filters applied until this point result in 2349 pairs of sets of instructions.

6.5.2 Conversion of Instructions into Protocols

After extracting sets of instructions, each of these was converted into a protocol $\langle T, \prec \rangle$ as described in Section 6.2. To do so, the steps and requirements of a set of instructions are interpreted as the elements of the set of tasks T. The ordering of the steps in the set of instructions is included in the dependencies $\prec$ of the derived protocol. For example, if a step s_1 precedes another step s_2, then $s_1 \prec s_2$ is added as a dependency. The dependencies of the protocol also specify when a certain requirement should be made available. For each requirement r of a set of instructions, a dependency $r \prec s$ is added, where s is the first step in the set of instructions that requires r. Following our example, the requirement of a "tea bag" is used for the first time in the step "adding the tea bag in the cup". Therefore, the dependency relation $\prec$ specifies that this requirement should be made available before that step.

6.5.3 Protocol Selection

The final phase of the data acquisition pipeline involves selecting pairs of protocols which are compatible with each other. To do this, we defined the *dependency tree* of a protocol $\langle T, \prec \rangle$ as the transpose of the graph which has T as the set of nodes and $\prec$ as the set of edges. This graph is a rooted tree with the task corresponding to the last step of the set of instructions as the root. The last step is chosen as the root because there is no other task that depends on it. This graph is a tree because the steps of the instructions, being in a total order, result in a single branch of the tree. Requirements result in additional branches of the tree. These branches are independent and do not result in cycles since no

requirement depends on another requirement or step.

If two rooted trees are not isomorphic, then there is no bijective translation τ between the nodes of the trees that is edge-preserving (Elgot et al. 1978). Therefore, if two protocols do not have isomorphic dependency trees, then they are not compatible, as there is no translation which can always map a successful execution in one protocol into another successful execution for the other protocol. Pairs of protocols which do not have isomorphic dependency trees were excluded from our experiment. After this selection, we obtained a final set of 327 pairs of protocols.

6.5.4 Label Cleaning

We processed natural language labels of tasks using the typical natural language pipeline. We first cleaned them, removing numbers, abbreviations, and punctuation marks other than dots. Of course, numbers would help to decide which is the correct mapping. We removed them because, although they are common in the cooking domain, they do not necessarily appear in all kinds of protocols.

We then performed a Part of Speech tagging to identify the grammatical function of each word in the sentence. We used the FreeLing natural language processing tool suite (Padró and Stanilovsky 2012), since it provided better results for Spanish than others. We kept the nouns, adjectives, verbs, and adverbs, and we used their lemmas, that were automatically obtained with the same tool. For example, the original sentence *Get 2 cups of hot water* was transformed into *get cup hot water*.

6.5.5 Identification of the Semantically Correct Translation

A pair of protocols usually had more than one possible isomorphism. That implies that they are compatible under more than one translation. As we mentioned before, we implicitly assumed that, among all the translations under which a pair of protocols is compatible, there exists one that is semantically correct. Although we did not check this was the case for all the protocols, it seemed to be true very frequently, if not always.

For each pair of protocols we computed a translation that we identified as the one most likely to be semantically correct. To this end we used a combination of techniques, that included dataset-specific properties in the Human Instruction Dataset, such as the structure of the instructions and the specific classification of tasks (i.e. steps, tools or ingredients). We also used different external semantic resources such as online translators. We made final decisions manually when there was a conflict or when the semantically correct alignment was not easily identified.

6.6 Evaluation

We evaluated the performance of the proposed techniques in two different ways. First, we studied how useful these techniques are to let agents interact meaningfully, by analyzing the success of their executions when using the translations obtained with them. Second, we evaluated directly the quality of the translations obtained from ω, by comparing them to external resources. In this case, we adopted three evaluation techniques: an automatic evaluation using an existing dictionary as a reference, a manual expert evaluation, and a crowdsourcing experiment.

6.6.1 Success Rate

We first analyzed how well agents can interact using the translations that they learn with our technique. We used the notion of semantic success, that determines when joint executions are successful under the semantically correct translation between tasks. Concretely, a joint execution is semantically successful if it is successful in both protocols when translated with the semantically correct translation. An experiment consists on a training phase, during which agents perform a fixed number of training interactions following randomly chosen protocols. During these interactions agents updated their alignment ω as explained before. Then, we performed 100 test interactions, without updating, computing the local translation from the learned ω. We measured how many of the test interactions were successful. We performed the experiment for 2^n training interactions, with n between 0 and 11. For each number of training interactions, the experiment was repeated 5 times for 5 different sets of training and test protocols.

We compared two different strategies to compute δ:

- the basic agent described in the main part of 6.4.1;

- the basic agent, without taking frequencies into account.

We compared these agents with the rate of success obtained when agents use an external translation, which was extracted from the Oxford Dictionary Spanish-English (`https://es.oxforddictionaries.com/english-spanish`, accessed on 21-02-2018) translation module. Agents using the Oxford Dictionary choose their interpretations using δ as we described, but instead of learning a distribution over word mappings δ, they use one extracted from the dictionary as follows. For foreign and local words v_2 and v_1 respectively,

$$\omega(v_1, v_2) = \begin{cases} 1 & \text{if } v_1 \text{ is a translation for } v_2 \text{ in the dictionary} \\ 0 & \text{otherwise} \end{cases}$$

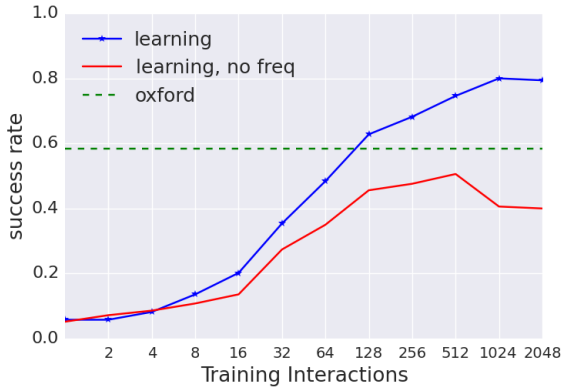


Figure 6.1: Success rate for different number of training interactions. *learning* agents use the technique that we propose, *learning no freq* agents do not take into account the frequency of words, and *oxford* agents use the Oxford Dictionary translation, without learning.

Figure 6.1 shows the success rate for the three agents. First, note that agents that take into account word frequencies perform better than those who do not. In particular, agents that do not consider frequencies become worse after many interactions, when seeing some words much more often than others starts to affect the values.

Our alignment technique allows agents to collaborate successfully in nearly 80% of the cases, and they outperform the success rate obtained with the dictionary after only around 100 interactions. These results are obtained with very simple updating techniques and no semantic resources at all. Interestingly, a translation that is semantically correct can be obtained from an update that only takes into account structural properties. Even if a pair of protocols have many compatible translations, our technique will find the one that is semantically correct. This is because agents learn from many pairs of protocols and not only one, and the correct one is the only translation that makes all pairs compatible. *Get a tea bag* may be mapped with *microwave water* in one protocol, but the translation will not work for others. To confirm this, we measured the success rate obtained with the original interaction-based alignment method from Chapters 3 and 4, that is, using a confidence distribution between sentences directly. The results were always close to 0, because agents do not choose the alignment that is semantically correct.

The success rate does not converge to 100%. This is because some protocols have translations that are difficult to learn. For example, one of the protocols pairs has *cuchara de helado* (ice cream spoon) corresponding to *melon baller*, even when their individual words are not similar. As we discuss later, an ap-

proach considering noun phrases would solve this issue. In other cases, the protocols had labels that only differed on a number, such as a protocol calling for *1/2 teaspoon salt* and *1 teaspoon salt*. Since we simplified the labels by removing numbers, these two tasks have the same representation, and therefore become indistinguishable to our agents. These cases, however, are not very frequent. Other protocols had misspelled or strangely written words that lead to confusion. For example, a protocol called for a *tea spoon* (instead of a *teaspoon*). The agent mapped this task to *té* (tea).

6.6.2 Translation Quality

The main objective of this work is to learn a translation between words which increases the success rate of the interactions between the agents. However, it is reasonable to expect that a useful alignment would map together words which are the translations of each other. In what follows we investigate whether this is the case. This task may seem simple, since ω is an alignment between words in existing languages. However, even when there exist multiple resources to obtain translations between English and Spanish, we will argue that finding a useful one is not simple.

To evaluate the translations, it is first necessary to define how these are obtained from the confidence distribution ω. In the techniques, agents build local translations by choosing a mapping between those that have more confidence. Now we do not need to consider only one mapping, but all those that have higher confidence.

Definition 6.5. Let ω be the alignment computed by a_1. A *relational translation* between words in V_1 and V_2 is a relation $Tr : V_1 \times V_2$ that contains all mappings with higher value. For $v_1 \in V_1$, $v_2 \in V_2$,

$$(v_1, v_2) \in Tr \Leftrightarrow v_1 \in \underset{v \in V_1}{\operatorname{argmax}}\ \omega(v, v_2)$$

■

Note that this one is the relational translation that can be obtained from the alignment computed by a_1, which has many translations for a $v_2 \in V_2$. The one for a_2 is computed analogously.

We obtained this set for the confidence distributions of both agents, getting an English-Spanish translation from the agent using Spanish protocols, and a Spanish-English translation from the other one. In the following subsections we discuss two ways of determining the quality of Tr.

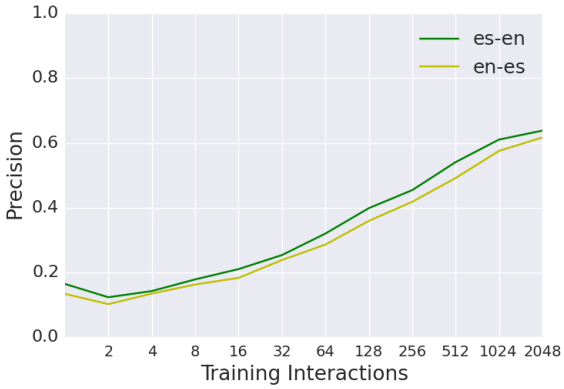


Figure 6.2: Precision against the Oxford Dictionary reference

Oxford Dictionary

We first evaluated automatically the translations that we obtained, using the Oxford Dictionary as a reference. We used the standard *precision* measure (see Definition 3.9). This measure is enough here, since we are interested in analyzing how many of our translations are correct.

Of course, the problem when evaluating a translation by computing its precision relies on finding a good reference. This is particularly important since mappings that are not in the reference are considered incorrect. In our case, we found that many of the words used in the protocols were not in the Oxford Dictionary. In many cases, this happened with words in languages other than Spanish or English. Particularly in the cooking domain, many expressions are used in their original form, such as *roti*, *bouillon*, *dashi*. Other words are informal (such as *carnitas*), or were misspelled by the humans who wrote the labels.

To take this into account and avoid evaluating mappings between words that do not even appear in the translation, we built a subset of *evaluable* translations. For each translation (English-Spanish and Spanish-English) we computed a reference translation and a translation to evaluate. Let us briefly explain the process with the Spanish-English case. We first obtained all the Spanish and English words in the protocols that the agents had seen, that we call V_{es} and V_{en} respectively. Let Ox be the complete Oxford translation between Spanish and English and Tr the translation obtained by the agent. The reference Ox_R are those mappings $(v_{es}, v_{en}) \in Ox$ such that $v_{es} \in V_{es}$ and $v_{en} \in V_{en}$. The translations to evaluate, $Trans_{Ev}$, where those $(v_{es}, v_{en}) \in Tr$ such that $(v_{es}, v) \in Ox_R$ and $(v', v_{en}) \in Ox_R$ for some $v \in V_{en}$, $v' \in V_{es}$. The idea behind this is to only evaluate mappings between words for which the dictionary has

the correct translation. Table 6.1 shows the sizes of the translations we obtained for agents that had interacted 2048 times. As we can see, only around 63% of the total mappings can be evaluated in each case. The Oxford Dictionary has many more mappings than our translation, even between words in the protocols. These are, in general, mappings that do not correspond to the cooking domain, or that are not commonly used. For example, it mapped *thicken* with *crecer* (to grow), which is an uncommon translation for the context of a recipe.

	Original	Evaluable	Oxford (Ref)
es → en	1908	1209	2047
en → es	1948	1280	2416

Table 6.1: Translation sizes after 2048 training interactions

Figure 6.2 shows the precision of the evaluable portion of the obtained translations with respect to Ox_R after different training sessions The precision starts already with a value of approximately 20%. This is because the translation Tr has very few mappings when the agents have only interacted once. After a few interactions it decreases, since the agents are considering more words without having experience about them yet. The precision then increases with the number of interactions, but less sharply than the rate of successes. We will discuss this in more detail at the end of this section.

Expert-Based Evaluation

The numbers in Table 6.1 suggest that the Oxford Dictionary may not be the best reference to evaluate a domain-specific translation like the one we obtain. In this section we show a human-based evaluation, in which the author of this dissertation manually evaluated a subset of the mappings. The author of this dissertation is considered to be an expert because she speaks fluent English and native Spanish, and has extensive experience reading and following recipes in both languages. We are aware that this is not a recommended practice, and that results should be evaluated by external experts. For this reason we performed a different evaluation that we discuss next. The expert evaluated part of a translation obtained after 1500 training interactions, around where the improvement appears to have reached a plateau (see Figure 6.2). The evaluation was performed over a random sample of 200 mappings for each translation.

The results of this evaluation are displayed in Table 6.2. The precision remains in the same levels as before when all mappings are taken into account, and in fact it is a bit higher. Since this evaluation is performed in all the mappings, this shows that there are many correct mappings that the Oxford

	Oxford Dict	Expert
es → en	0.616	0.66
en → es	0.578	0.625

Table 6.2: Expert evaluation

Dictionary cannot find. In many cases this is because the words do not appear in the dictionary. The slightly higher precision for the manual evaluation could be an effect of choosing a sample of words to evaluate, but it can also mean that there are mappings that the Oxford Dictionary classifies incorrectly because they are domain-specific. For example, the dictionary labeled as wrong the mapping between *jar* and *frasco*, when they clearly refer to the same thing.

Crowdsourcing

Since the judgment of a single expert can be biased, we further tested the obtained results we obtained with a crowdsourcing experiment. To this aim, we set up a task in the CrowdFlower platform (`https://www.crowdflower.com`). The task was simple: we asked the participants whether two words in the same experiment could be mapped. We developed this experiment only for the translation from Spanish to English, to optimize the resources. Concretely, let *es* be an expression in Spanish, and *en* the mapping our technique obtained. We provided the participants with contextual information, since we wanted to identify whether mappings were correct in the context of a particular protocol. Let $task_{es}$ be a task where *es* appears, and $prot_{es}$ the name of the protocol where the task is. Concretely, we asked the participants to answer Yes or No to the following question:

> In the context of a set of instructions titled $prot_{es}$,
> can *en* refer to the same thing as *es*?
>
> For example, in $task_{es}$

We filtered participants by requiring them to speak both languages. To enforce this restriction, we only admitted participants from the United States or Spanish-speaking countries. We provided them with an explanation of the task and a set of *test questions*, that were useful to teach them how the task worked, as well as to filter participants with bad performance. We let each participant perform a maximum of 25 judgments.

We chose randomly a set of 400 mappings to evaluate. When the experiment finished, we had 2213 judgments for 219 mappings. Table 6.3 shows the results in terms of the precision, that is, the proportion of mappings that were classified as correct. We show the results for three different levels of agreement. The agreement on the classification of a mapping is the proportion of participants that chose the final answer. For example, if a mapping with answer Yes has agreement 0.7, a 70% of the participants answered Yes and a 30% answered No. The column >= 0.5 has all the mappings, the other ones are filtered.

Agreement	>= 0.5	> 0.8	1
precision	0.626	0.653	0.685
number of items	219	196	178

Table 6.3: Crowdsourcing evaluation, for es → en data

The results for mappings with agreement higher than 0.8, that looses a 10% of the mappings, are similar to the ones found by the expert. However, we identified certain behaviors that suggest that our experiment may have been designed in a better way. For example, a 78% of the participants agreed that *brown* could not be mapped to *integral*, when it is actually a good translation when talking about rice. For some reason, they also did not agree on the correct mapping between *filtrar* and *strain*, or between *brine* and *salmuera*. They also mostly thought that *semisweet* was not *semiamargo*. This makes sense, because *sweet* is the opposite of *amargo*, but those are the words that are actually used. This raises two issues related to the experiment that we designed. First, the contextualization we provided seems not to be enough. We observed that mappings that were quite evident, but only correct in context were considered more often incorrect than those that were difficult translations, but context-independent. This could happen because the participants want to finish the task as fast as possible, and do not take the time to read the instruction we put as example. The second issue is the one of language proficiency. We found that it was necessary to be proficient not only in both languages (some words can be difficult for second-language speakers) but also in the cooking domain (even native speakers may not know the meaning of *sift*). While it is easy to find participants that have almost-native level in both English and Spanish, it is not so simple to make sure that they are familiar with the specific vocabulary that was used in the tasks. We think, for a future experiment, it would be better to have less participants, and focus on the familiarity with the domain.

Participants also said some words were correct mappings when they actually were not. In general this did not happen in the same word, so there were not many false positives. This may be because they were pressured to answer, and

when not knowing an answer they chose what they (correctly) thought would be better for the experiment. We think this could be solved by adding a *Skip* option to get another word, or a graded answer allowing them to choose how confident they are on the mapping of two words, although this would make the analysis of the results more difficult.

6.6.3 Evaluation Discussion

While the obtained translations are useful to make agents interact successfully, their semantic precision values are not sufficiently high to consider them reliable translations. At this stage, translations cannot be used for other purposes apart of interaction, such as to translate a recipe from scratch. This is due in large part to unresolved issues in the initial cleaning and natural language processing of the protocol labels and would be improved with a more efficient NLP tool, able to identify misspellings and to lemmatize words correctly.

The main issue, however, is not being able to identify noun phrases. A good example is the one of *xanthan gum*, and its Spanish equivalent *goma xantana*. These words appear always together in the protocol, so the agent has no way of identifying if the correct alingment is {(goma, gum), (xantana,xanthan)} or {(goma, xanthan), (xantana, gum)}. However, both mappings are useful to interact when the words appear together, which explains why the performance of the agents achieves better results than the translation's precision. Moreover, many concepts can be described with one word in a language but need more in another one. For example, *cornstarch* and *fécula de maíz*, or *turn off* and *apagar*. Naive word-to-word translations like the ones that we obtain will never translate these expressions correctly: *turn* is not semantically equivalent to *apagar*. This issue can be solved in two ways. One option is to use a more powerful processing that identifies noun phrases. The other one is to let agents identify them automatically. This, however, would require more training examples and much more computation time.

With these issues solved, our translations would be a very useful context-specific resource. On one side, we have already observed that a typical dictionary does not have many of the particular words that are commonly used in a jargon. In addition, general dictionaries provide many possible translations for a word, letting user needs to identify which one is useful for its needs. Our translations, instead, would directly provide a translation that is suitable for the context.

6.7 Conclusions

The results obtained in this chapter show that the abstract techniques that we proposed in earlier chapters can be adapted to be applied on real data. This adaptation, however, is not straightforward. This is, mainly, because natural

language protocols are labeled with sentences instead of atomic messages. These sentences are not commonly repeated exactly. For this reason, it is more convenient to maintain mappings between words and compute the ones for sentences from them. The technique we propose to do this, although very simple seems to work well enough. Moreover, it does not make strong assumptions on the languages, which yields a general technique. However, this generality comes at a price: our method is computational expensive, in particular when sentences are long. This is an aspect that should be improved in future work.

This is the first time that we evaluate our techniques from a really semantic point of view, analyzing if the mappings that are obtained are actually correct in the real world. We showed that agents get to perform semantically correct interactions, and we obtained a translation of decent quality. This is interesting, since the updating techniques consider the meaning of the tasks of the instructions, but only structural properties. We then show that, even when there may be different such translations, agents learn the one that is semantically correct, meaning the one that maps together equivalent tasks. This is because we consider agents that engage in collaborations across multiple protocol pairs, and the semantically correct translations are the most useful to them to achieve the highest rate of compatibility across those protocols.

Our techniques allow agents to learn how to collaborate completing of protocols without using external resources. In fact, agents that use our technique outperform those that use the translations from a well-known dictionary to obtain translations. This suggests that, in addition to scenarios where external resources are not available or expensive to use, our method can be used to complement approaches based on external resources to discover additional mappings between words, such as domain specific ones.

Chapter 7

Conclusion

Our objective in this dissertation was to explore the idea of grounding vocabulary alignment in procedural knowledge uncovered in interaction. We investigated how agents can learn an alignment between their local vocabularies and foreign ones, or even an entire semantics, by observing how interactions develop. This type of adaptation is one humans are very good at, but it had not been studied in depth before for artificial agents. Its main advantage is that it does not assume the availability of any external resources other than task specifications, which are necessary to interact even for agents that share a language.

Along the four main chapters of this dissertation we extended the *interaction-based vocabulary alignment* technique, first proposed in (Atencia and Schorlemmer 2012) and (Atencia 2010), applying it to different types of protocol representations and combining it with external knowledge. Concretely, we considered agents that learn an alignment from interacting when they share the knowledge of how to perform tasks specified with finite-state machines (Chapter 3), constraint-based protocols (Chapter 4) and partial orders (Chapter 6). In Chapter 5 we used commitment specifications, but solved a different, although related, problem. There we studied how agents can learn specifications from observing interactions between other agents who already know them.

In all cases, we proposed methods that are based on probabilistic techniques. This allows us to handle the intrinsic uncertainty of the problem, which arises from the fact that an interpretation builds up sequentially, so misunderstanding one message can cause problems when interpreting future ones. The updating techniques that we used are inspired on well-known learning approaches such as Reinforcement Learning in Chapter 3 or Bayesian updating in Chapter 4. In this way, we were able to frame our work in the context of extensively investigated areas, and to adapt their ideas to our case of interest.

The methods we propose embody a particular take on the question of what determines the *meaning* of a term. We do not consider that meaning is given by

reference to an object in the world, nor by a definition composed of other words, like it was commonly done in previous work. Instead, the meaning of a word is viewed as determined by the effects that its utterance has in an interaction. Of course, this affects the notion of equivalence in an interaction: now two words are equivalent if they can be used to the same effect.

In accordance to this view of meaning, the objective of the techniques that we propose here is not to learn a *correct* alignment, but to find one that is useful to let agents interact successfully. We used this idea explicitly to evaluate the performance of the methods in Chapters 3 and 6, measuring how frequently agents' interactions succeed when they use alignments that they learned with our methods. In Chapter 4, instead, we used more traditional alignment evaluation techniques: comparing the ones obtained by the agents to a reference. However, the spirit of the evaluation remains the same, since this reference is defined as that one under which all protocols are compatible, that is, the one that is useful to interact. In Chapter 5 we compared these two evaluation methods and showed that, because of how the commitment semantics are defined, they were not the same. In all cases, the evaluation showed that the techniques that we propose (at least the more sophisticated ones) are useful to find a translation that agents can use to interact with each other successfully.

In Chapters 3, 4, and 5 we evaluated our techniques on a set of randomly generated datasets. This allowed us to cover different levels of protocol complexity, avoiding the biases introduced by specific datasets. This general evaluation showed that the approaches work in abstract. In Chapter 6 we adapted the techniques to be applied on real data. Concretely, we used a set of human-crafted protocols extracted from the WikiHow webpage. This allowed us to verify that the technique can also be useful for real data, although some non-trivial changes had to be made to consider the particularities of natural language. In particular the fact that messages are sentences and not just words.

The evaluation of our methods showed that they are useful when many interactions are performed or observed. This is necessary even for the small vocabularies that we used. This may put in doubt the direct applicability of our technique to a real case. However, it is important to keep in mind that, in real situations, the context of interaction is rarely the only source of meaning that is available. Speakers can usually rely on other aspects, such as syntactic similarity or visual information, or they have a meta-language that they can use to ask for help. The following are some of the advantages of using the interaction-based approach.

1. It provides a simple way to repair and enhance existing knowledge. As we showed, our techniques are fast when there is a good previous alignment.

2. It obtains information in a lightweight way, by exploiting a source that

was overlooked before and that is free to agents that collaborate: the knowledge of how to interact.

3. It is useful to learn *how to use* a vocabulary. Interaction specifications have information about the meaning of words that may be missing in other sources. For example, the problem of meaning disambiguation, common in other techniques, does not exist in our case.

Summarizing, in this dissertation we showed how an aspect of meaning that had not been considered before can be useful to learn vocabularies in artificial environments, and we provided techniques to do this. Importantly, these techniques do not make strong assumptions about the situation in which they can be used; can be combined with other sources of meaning; and can be adapted to an empirical dataset. Let us now discuss four aspects that we consider to be particularly interesting, together with their related future work.

7.1 Differences between Protocol Specifications

A natural question derived from this work is that of how the specifications techniques we investigated compare when considering interaction-based vocabulary alignment. What are the differences of the learning techniques when applied to each of them? Is there a particular one that makes inference simpler?

Since each specification technique expresses a different type of knowledge, it is difficult to use them to define protocols with the same information. Therefore, it is also complicated to perform an experiment to determine which specifications are best in terms of how fast agents learn useful alignments when using them. However, we did observe some important differences that we comment on now.

A first important aspect to discuss is the quantity of information described in each type of protocol. Transition-based protocols have much more information than constraint-based ones, and also more than the partial orders that we proposed in Chapter 6. This is because in each state there are only a few possible messages. In open protocols, instead, everything is possible unless constraints say otherwise, and the same is true of partial orders. This increase in information comes at the cost of a much more restrictive specification style, which explicitly describes all possible interaction flows. We used two different conceptions of alignment for these types of protocols. For transition-based ones, we considered alignments to be parametrized by states, which defines how each word has to be translated at each particular moment in the interaction. This can be done because there is enough information to define exactly what each word means at each point. For the other two specification formalisms we considered a cross-situational technique that learns general mappings, which are assumed to be useful for different situations. While the first way of representing alignments

can capture the contextual aspect of meaning in a better way, it yields a method in which the learning process is useful for only one protocol.

Another aspect to discuss is the uncertainty about which mappings are correct and which ones are not that each type of protocols produces. In constraint-based protocols it is clearly easier to learn from interactions that fail. When following transition-based protocols, agents only realize something went wrong in the interaction when they finish it. At that point, they cannot know which word in the sequence was mapped incorrectly, which is known as the *delayed reward* problem in the Reinforcement Learning literature. Open protocols do not have this problem; non-p-necessary constraints are immediately recognized as violated, and they only depend on two mappings, so there is less uncertainty about which interpretation could have been wrong. It is true that, inversely, when an interaction is successful in a transition-based protocol, agents can be sure of having made the correct mappings, while this is not the case for those based on constraints. However, it is generally very difficult to reach a successful interaction in the first type of specification.

7.2 Combination with Other Sources of Meaning

In Chapters 3 and 4 we discussed how external alignments can be incorporated in an interaction-based alignment technique explicitly. In both cases, the integration is achieved by considering external alignments as prior knowledge in the learning process. That is, the initial values in the alignment that represents the agent's confidences on mappings, otherwise extracted from a uniform distribution, are obtained from the previous alignments. These two options parallel the well known debate that opposes the Bayesian and frequentist interpretations of probability (see for example Carlin and Louis (1997)). A frequentist approach considers all alignments as equally possible until it finds out which one works well more often. A Bayesian approach incorporates external knowledge, even at the risk that it is incorrect.

In these two chapters, we measured how the quality of external alignments affected the learning process, observing an interestingly phenomenon. In the case of finite state automata, the best alignments were those that have large recall. The level of precision, although not negligible, is not nearly as relevant as that of recall. In the case of open protocols the situation was exactly the opposite: precision was the most relevant value, while recall had almost no effect, or even a negative effect. This is because of the fundamental differences in what both methods express, and they can be explained by how words are selected on each case. As we explained before, low recall means that many of the correct mappings were not found, while low precision means that many incorrect mappings were found. In the transition-based case, agents have to choose inter-

pretations between a few possibilities. If there is an incorrect mapping in the alignment, it is unlikely that it is between a received word and one of the ones being expected. For this reason, having many incorrect mappings is not a problem, and it is preferable to have many correct ones. In open protocols, instead, having incorrect mappings is problematic, since the agent choose between many possible words. The notions of *misleading* and *useful* mappings that we propose at the end of Chapter 3 capture these ideas. Of course, this does not take into account how expected words are semantically related, and how this could affect how they are mapped. Relating protocols with alignments semantically is, as we will discuss soon, one of the directions of future work derived from this thesis.

Although we have not explored how the techniques in Chapters 5 and 6 perform when combined with external alignments, the integration would be very similar to the one proposed in the preceding two chapters. In particular, we think that the technique for the empirical data could be good in combination with existing translation techniques, for example to find missing or domain-specific mappings.

Until now, we only considered external alignments to be a confidence distribution over possible mappings, without any semantic connection to the interaction protocols. It would be very interesting to consider a more meaningful integration between our techniques and external knowledge resources. However, to that aim it is first necessary to define a semantic relation between the information in the protocols and that in the external resources, which still does not exist. We proposed a first draft of this idea in (Chocron and Schorlemmer 2016b), where we present an idea for a protocol specification language that uses ontological knowledge. For example, our language can be used to say *if the waiter asks to the customer what she wants to drink, the customer will answer specifying a drink* in the following way:

$$utter(W, C, \mathsf{Drink}) \Rightarrow X\ utter(C, W, \mathsf{D}) \wedge \mathsf{D} \sqsubseteq \mathsf{Drink}$$

where Drink is an ontological class, C and W are the customer and waiter respectively, X is the *next* operator described in Chapter 4, and *utter* is a predicate that is true when an agent sends a message. In this way, the alignments between ontologies could be meaningfully used while performing an interaction-based technique. The language we propose in (Chocron and Schorlemmer 2016b), however, is very preliminary.

7.3 Non-Compatible Protocols

During most of this dissertation we assumed that interlocutors have compatible protocols. That means, in general, that their specification accepts the same interactions (modulo a translation) as correct. This was a useful assumption to

focus on vocabulary alignment grounded on common knowledge about how to perform tasks. However, as we discussed in Chapter 4, it is a strong restriction in practical terms.

First of all, it is important to note that the fact that we work with probabilistic techniques means that our methods should be robust against small differences in the protocols, in particular in the cross-situational approach. In this case, a pair of non-compatible protocols will produce interactions that can never be finished correctly, but this situation can be considered as noise if it is not too frequent, and agents should be able to learn anyway. In the transition-based protocols the situation is different, because there is only one protocol. In this case, the best the technique can do is aim to learn mappings for the part of the protocol that can actually be mapped.

In Chapter 4 we discussed an extension in which protocols have constraints associated to weights. The weight of a constraint represents the punishment received by agents when it is violated. We showed that agents can use the interaction-based alignment ideas to find an alignment when their specifications differ in a small part and, interestingly, to find an alignment that minimizes the punishment even if their protocols are not similar. We think this last point has potential to be further explored. It involves an interesting premise: there is no correct alignment, but better and worse ones. We think it is possible to investigate how different types of protocols affect this dynamic, as well as how they change for different types of agents, who can care more or less about violating constraints.

7.4 Relation with Grammar and Language Structure

In most of this work we assume that messages are single words, which is perhaps the most unrealistic of our restrictions. Grammatical structure provides an enormous flexibility to language, which is lost when considering words atomically. In fact, it is only reasonable to consider messages without any structure when the scenarios are small and static, which are not the situations that motivate our techniques.

The lack of a way of approaching grammatical structure was the main obstacle we found when adapting our techniques to a set of empirical, human-crafted protocols with labels in natural language in Chapter 6. We discussed why it is still important to use alignments between words, and showed how these alignments can be used to translate sentences. In this case we considered sentences to be just bags of words. Despite being a very simple idea, we showed that it is useful to obtain good results, and it makes no strong assumption about the properties of language. It has, however, three drawbacks. First, the method that we proposed is computationally very slow, since it considers all possible

permutations of words. Second, it does not consider that the same concepts can be explained with different words. Finally, it does not attempt to exploit the compositional structure of natural language, which may substantially aid translation.

We think that it is important to address the inclusion of structure from two points of view. The first one is to consider how to map complex constructions. One of the directions that we planned to investigate but still remains open consists in considering agents that send first-order expressions instead of words in a propositional language, and we planned to analyze how this structure could be harnessed. We think that, for the applications that we have in mind, it is important to consider natural language messages, as we did in Chapter 6. An important objective for the future is to develop these methods. However, in some cases it may not be evident how sentences relate with the effects in an interaction. For example, when considering commitment specifications, it is not clear how different commitment operations would be combined in a sentence. It is necessary to first develop these relations to be able to consider structured messages. A related problem to consider is the one of constraints that have logical expressions instead of just one message (for example, *if you say beer AND chips then you cannot ask for coffee*).

The second point of view, which we have not developed here, consists in using our technique to learn the grammatical structure of a foreign language. In this way agents would not only infer a vocabulary alignment, but by observing patterns they could also obtain information about how the foreign language is structured. A simple version of such grammar induction could significantly enhance the techniques in Chapter 6. There, a problem was that agents could sometimes not find an alignment because they map words with each other, and sometimes concepts are described with more than one word in one language and only one in the other one. The question, particular to natural language, is whether agents can learn multi-word expressions with our techniques. Preliminary experiments indicate that, while it could be possible, it is computationally very expensive, since there are many words that could go together.

Bibliography

Karl Aberer, Philippe Cudré-Mauroux, and Manfred Hauswirth. The chatty web: emergent semantics through gossiping. In *Proceedings of the Twelfth International World Wide Web Conference, WWW 2003, Budapest, Hungary, May 20-24, 2003*, pages 197–206, 2003. doi: 10.1145/775152.775180.

Andrea Addis and Daniel Borrajo. *Information Retrieval and Mining in Distributed Environments*, chapter From Unstructured Web Knowledge to Plan Descriptions, pages 41–59. Studies in Computational Intelligence. Springer Verlag, 2010.

Eneko Agirre, Carmen Banea, Daniel M. Cer, Mona T. Diab, Aitor Gonzalez-Agirre, Rada Mihalcea, German Rigau, and Janyce Wiebe. Semeval-2016 task 1: Semantic textual similarity, monolingual and cross-lingual evaluation. In *Proceedings of the 10th International Workshop on Semantic Evaluation, SemEval@NAACL-HLT 2016, San Diego, CA, USA, June 16-17, 2016*, pages 497–511, 2016.

Nirav Ajmeri, Jiaming Jiang, Rada Chirkova, Jon Doyle, and Munindar P. Singh. Coco: Runtime reasoning about conflicting commitments. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, IJCAI 2016, New York, NY, USA, 9-15 July 2016*, pages 17–23, 2016.

Dana Angluin and Leonor Becerra-Bonache. A model of language learning with semantics and meaning-preserving corrections. *Artif. Intell.*, 242:23–51, 2017. doi: 10.1016/j.artint.2016.10.002.

Manuel Atencia. *Semantic alignment in the context of agent interaction.* Doctoral dissertation, Universitat Autònoma de Barcelona, 2010.

Manuel Atencia and W. Marco Schorlemmer. Formalising interaction-situated semantic alignment: The communication product. In *International Symposium on Artificial Intelligence and Mathematics, ISAIM 2008, Fort Lauderdale, Florida, USA, January 2-4, 2008*, 2008.

Manuel Atencia and W. Marco Schorlemmer. An interaction-based approach to semantic alignment. *J. Web Sem.*, 12:131–147, 2012. doi: 10.1016/j.websem.2011.12.001.

J.L. Austin. *How To Do Things With Words*. Oxford University Press, sep 1975. doi: 10.1093/acprof:oso/9780198245537.001.0001.

Franz Baader, Diego Calvanese, Deborah L. McGuinness, Daniele Nardi, and Peter F. Patel-Schneider, editors. *The Description Logic Handbook: Theory, Implementation, and Applications*. Cambridge University Press, 2003.

Matteo Baldoni, Cristina Baroglio, and Elisa Marengo. Behavior-oriented commitment-based protocols. In *ECAI 2010 - 19th European Conference on Artificial Intelligence, Lisbon, Portugal, August 16-20, 2010, Proceedings*, pages 137–142, 2010a.

Matteo Baldoni, Cristina Baroglio, and Elisa Marengo. Behavior-oriented commitment-based protocols. In *ECAI 2010 - 19th European Conference on Artificial Intelligence, Lisbon, Portugal, August 16-20, 2010, Proceedings*, pages 137–142, 2010b.

Matteo Baldoni, Cristina Baroglio, Elisa Marengo, and Viviana Patti. Constitutive and regulative specifications of commitment protocols: A decoupled approach. *ACM TIST*, 4(2):22:1–22:25, 2013. doi: 10.1145/2438653.2438657.

Mihai Barbuceanu and Mark S. Fox. COOL: A language for describing coordination in multi agent systems. In *Proceedings of the First International Conference on Multiagent Systems, June 12-14, 1995, San Francisco, California, USA*, pages 17–24, 1995.

Andrea Baronchelli, Maddalena Felici, Vittorio Loreto, Emanuele Caglioti, and Luc Steels. Sharp transition towards shared vocabularies in multi-agent systems. *Journal of Statistical Mechanics: Theory and Experiment*, 2006(06): P06014–P06014, jun 2006. doi: 10.1088/1742-5468/2006/06/p06014.

Marco Baroni and Alessandro Lenci. Distributional memory: A general framework for corpus-based semantics. *Computational Linguistics*, 36(4):673–721, 2010. doi: 10.1162/coli_a_00016.

Samuel Barrett, Noa Agmon, Noam Hazon, Sarit Kraus, and Peter Stone. Communicating with unknown teammates. In *ECAI 2014 - 21st European Conference on Artificial Intelligence, 18-22 August 2014, Prague, Czech Republic - Including Prestigious Applications of Intelligent Systems (PAIS 2014)*, pages 45–50, 2014. doi: 10.3233/978-1-61499-419-0-45.

Paolo Besana, Dave Robertson, and Michael Rovatsos. Exploiting interaction contexts in P2P ontology mapping. In *Proceedings of the 2005 International Workshop on Description Logics (DL2005), Edinburgh, Scotland, UK, July 26-28, 2005*, 2005.

Paolo Bouquet, Jérôme Euzenat, Enrico Franconi, Luciano Serafini, Giorgos Stamou, and Sergio Tessaris. Specification of a common framework for characterizing alignment. Deliverable D2.2.1, Knowledge Web, 2004.

Daniel Brand and Pitro Zafiropulo. On communicating finite-state machines. *J. ACM*, 30(2):323–342, 1983. doi: 10.1145/322374.322380.

Bradley P. Carlin and Thomas A. Louis. BAYES AND EMPIRICAL BAYES METHODS FOR DATA ANALYSIS. *Statistics and Computing*, 7(2):153–154, 1997. doi: 10.1023/A:1018577817064.

Andrew Carlson, Justin Betteridge, Bryan Kisiel, Burr Settles, Estevam R. Hruschka Jr., and Tom M. Mitchell. Toward an architecture for never-ending language learning. In *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2010, Atlanta, Georgia, USA, July 11-15, 2010*, 2010.

Justine Cassell. Embodied conversational interface agents. *Commun. ACM*, 43 (4):70–78, 2000. doi: 10.1145/332051.332075.

Michelle Cheatham, Zlatan Dragisic, Jérôme Euzenat, Daniel Faria, Alfio Ferrara, Giorgos Flouris, Irini Fundulaki, Roger Granada, Valentina Ivanova, Ernesto Jiménez-Ruiz, Patrick Lambrix, Stefano Montanelli, Catia Pesquita, Tzanina Saveta, Pavel Shvaiko, Alessandro Solimando, Cássia Trojahn dos Santos, and Ondrej Zamazal. Results of the ontology alignment evaluation initiative 2015. In *Proceedings of the 10th International Workshop on Ontology Matching collocated with the 14th International Semantic Web Conference (ISWC 2015), Bethlehem, PA, USA, October 12, 2015.*, pages 60–115, 2015.

Paula Chocron and Marco Schorlemmer. Attuning ontology alignments to semantically heterogeneous multi-agent interactions. In *ECAI 2016 - 22nd European Conference on Artificial Intelligence, 29 August-2 September 2016, The Hague, The Netherlands - Including Prestigious Applications of Artificial Intelligence (PAIS 2016)*, pages 871–879, 2016a. doi: 10.3233/978-1-61499-672-9-871.

Paula Chocron and Marco Schorlemmer. Interaction specifications as contexts for ontologies. In *Proceedings of the Joint Ontology Workshops 2016 Episode*

2: The French Summer of Ontology co-located with the 9th International Conference on Formal Ontology in Information Systems (FOIS 2016), Annecy, France, July 6-9, 2016., 2016b.

Paula Chocron and Marco Schorlemmer. Ontology alignment evaluation in the context of multi-agent interactions. In *Proceedings of the 11th International Workshop on Ontology Matching co-located with the 15th International Semantic Web Conference (ISWC 2016), Kobe, Japan, October 18, 2016.*, pages 25–36, 2016c.

Paula Chocron and Marco Schorlemmer. Vocabulary alignment for agents with flexible protocols. In *Proceedings of the Joint Ontology Workshops 2017 Episode 3: The Tyrolean Autumn of Ontology, Bozen-Bolzano, Italy, September 21-23, 2017.*, 2017a.

Paula Chocron and Marco Schorlemmer. Vocabulary alignment in openly specified interactions. In *Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems, AAMAS 2017, São Paulo, Brazil, May 8-12, 2017*, pages 1064–1072, 2017b.

Paula Chocron and Marco Schorlemmer. Inferring commitment semantics in multi-agent interactions. In *Proceedings of the 17th Conference on Autonomous Agents and MultiAgent Systems*, AAMAS '18 (to appear), 2018.

Amit K. Chopra and Munindar P. Singh. Constitutive interoperability. In *7th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2008), Estoril, Portugal, May 12-16, 2008, Volume 2*, pages 797–804, 2008. doi: 10.1145/1402298.1402335.

Amit K. Chopra and Munindar P. Singh. Generalized commitment alignment. In *Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2015, Istanbul, Turkey, May 4-8, 2015*, pages 453–461, 2015.

Amit K. Chopra, Alexander Artikis, Jamal Bentahar, Marco Colombetti, Frank Dignum, Nicoletta Fornara, Andrew J. I. Jones, Munindar P. Singh, and Pinar Yolum. Research directions in agent communication. *ACM TIST*, 4(2): 20:1–20:23, 2013. doi: 10.1145/2438653.2438655.

Alessandro Cimatti, Edmund M. Clarke, Enrico Giunchiglia, Fausto Giunchiglia, Marco Pistore, Marco Roveri, Roberto Sebastiani, and Armando Tacchella. Nusmv 2: An opensource tool for symbolic model checking. In *Computer Aided Verification, 14th International Conference, CAV 2002,Copenhagen, Denmark, July 27-31, 2002, Proceedings*, pages 359–364, 2002. doi: 10.1007/3-540-45657-0_29.

World Wide Web Consortium et al. RDF 1.1 primer. 2014.

R. Scott Cost, Ye Chen, Timothy W. Finin, Yannis Labrou, and Yun Peng. Using colored petri nets for conversation modeling. In *Issues in Agent Communication*, pages 178–192, 2000. doi: 10.1007/10722777_12.

Stephen Cranefield, Felipe Meneguzzi, Nir Oren, and Bastin Tony Roy Savarimuthu. A bayesian approach to norm identification. In *ECAI 2016 - 22nd European Conference on Artificial Intelligence, 29 August-2 September 2016, The Hague, The Netherlands - Including Prestigious Applications of Artificial Intelligence (PAIS 2016)*, pages 622–629, 2016. doi: 10.3233/978-1-61499-672-9-622.

John Dewey. *Logic: The Theory of Inquiry*. Henry Holt, 1938.

Mark d'Inverno, Michael Luck, Pablo Noriega, Juan A. Rodríguez-Aguilar, and Carles Sierra. Communicating open systems. *Artif. Intell.*, 186:38–94, 2012. doi: 10.1016/j.artint.2012.03.004.

Zlatan Dragisic, Kai Eckert, Jérôme Euzenat, Daniel Faria, Alfio Ferrara, Roger Granada, Valentina Ivanova, Ernesto Jiménez-Ruiz, Andreas Oskar Kempf, Patrick Lambrix, Stefano Montanelli, Heiko Paulheim, Dominique Ritze, Pavel Shvaiko, Alessandro Solimando, Cássia Trojahn dos Santos, Ondrej Zamazal, and Bernardo Cuenca Grau. Results of the ontology alignment evaluation initiative 2014. In *Proceedings of the 9th International Workshop on Ontology Matching collocated with the 13th International Semantic Web Conference (ISWC 2014), Riva del Garda, Trentino, Italy, October 20, 2014.*, pages 61–104, 2014.

Umberto Eco. *Experiences in translation*. University of Toronto Press, 2008.

Calvin C. Elgot, Stephen L. Bloom, and Ralph Tindell. On the algebraic atructure of rooted trees. *J. Comput. Syst. Sci.*, 16(3):362–399, 1978. doi: 10.1016/0022-0000(78)90024-7.

Jérôme Euzenat. Semantic precision and recall for ontology alignment evaluation. In *IJCAI 2007, Proceedings of the 20th International Joint Conference on Artificial Intelligence, Hyderabad, India, January 6-12, 2007*, pages 348–353, 2007.

Jérôme Euzenat. First experiments in cultural alignment repair (extended version). In *The Semantic Web: ESWC 2014 Satellite Events - ESWC 2014 Satellite Events, Anissaras, Crete, Greece, May 25-29, 2014, Revised Selected Papers*, pages 115–130, 2014. doi: 10.1007/978-3-319-11955-7_10.

Jérôme Euzenat. Interaction-based ontology alignment repair with expansion and relaxation. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017, Melbourne, Australia, August 19-25, 2017*, pages 185–191, 2017. doi: 10.24963/ijcai.2017/27.

Jérôme Euzenat and Pavel Shvaiko. *Ontology Matching, Second Edition*. Springer, 2013.

Jérôme Euzenat, Christian Meilicke, Heiner Stuckenschmidt, Pavel Shvaiko, and Cássia Trojahn dos Santos. Ontology alignment evaluation initiative: Six years of experience. *J. Data Semantics*, 15:158–192, 2011. doi: 10.1007/978-3-642-22630-4_6.

Timothy W. Finin, Richard Fritzson, Donald P. McKay, and Robin McEntire. KQML as an agent communication language. In *Proceedings of the Third International Conference on Information and Knowledge Management (CIKM'94), Gaithersburg, Maryland, November 29 - December 2, 1994*, pages 456–463, 1994. doi: 10.1145/191246.191322.

Michael C Frank, Noah D Goodman, and Joshua B Tenenbaum. Using speakers referential intentions to model early cross-situational word learning. 20:578–85, 05 2009.

Gottlob Frege. Sense and reference. *The philosophical review*, 57(3):209–230, 1948.

Laura Giordano, Alberto Martelli, and Camilla Schwind. Specifying and verifying interaction protocols in a temporal action logic. *J. Applied Logic*, 5(2):214–234, 2007. doi: 10.1016/j.jal.2005.12.011.

Yoav Goldberg and Omer Levy. word2vec explained: deriving mikolov et al.'s negative-sampling word-embedding method. *CoRR*, abs/1402.3722, 2014.

Claudia V. Goldman, Martin Allen, and Shlomo Zilberstein. Learning to communicate in a decentralized environment. *Autonomous Agents and Multi-Agent Systems*, 15(1):47–90, 2007. doi: 10.1007/s10458-006-0008-9.

Akin Günay, Michael Winikoff, and Pinar Yolum. Dynamically generated commitment protocols in open systems. *Autonomous Agents and Multi-Agent Systems*, 29(2):192–229, 2015. doi: 10.1007/s10458-014-9251-7.

Sandra Heiler. Sematic interoperability. *ACM Comput. Surv.*, 27(2):271–273, 1995. doi: 10.1145/210376.210392.

Richard J Herrnstein. On the law of effect. *Journal of the experimental analysis of behavior*, 13(2):243–266, 1970.

C. A. R. Hoare. *Communicating Sequential Processes.* Prentice-Hall, 1985.

Laura Hollink, Mark van Assem, Shenghui Wang, Antoine Isaac, and Guus Schreiber. Two variations on ontology alignment evaluation: Methodological issues. In *The Semantic Web: Research and Applications, 5th European Semantic Web Conference, ESWC 2008, Tenerife, Canary Islands, Spain, June 1-5, 2008, Proceedings*, pages 388–401, 2008. doi: 10.1007/978-3-540-68234-9_30.

John E. Hopcroft and Jeffrey D. Ullman. *Introduction to Automata Theory, Languages and Computation.* Addison-Wesley, 1979.

Michael Huth and Mark Dermot Ryan. *Logic in computer science - modelling and reasoning about systems (2. ed.).* Cambridge University Press, 2004.

Nancy Ide and Jean Véronis. Introduction to the special issue on word sense disambiguation: The state of the art. *Computational Linguistics*, 24(1):1–40, 1998.

Ningsheng Jian, Wei Hu, Gong Cheng, and Yuzhong Qu. Falconao: Aligning ontologies with falcon. In *Integrating Ontologies '05, Proceedings of the K-CAP 2005 Workshop on Integrating Ontologies, Banff, Canada, October 2, 2005*, 2005.

Chloé Kiddon, Ganesa Thandavam Ponnuraj, Luke Zettlemoyer, and Yejin Choi. Mise en place: Unsupervised interpretation of instructional recipes. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*, pages 982–992, 2015.

Andrew Koster, Jordi Madrenas-Ciurana, Nardine Osman, W. Marco Schorlemmer, Jordi Sabater-Mir, Carles Sierra, Dave De Jonge, Angela Fabregues, Josep Puyol-Gruart, and Pere Garcia-Calvés. u-help: Supporting helpful communities with information technology. In *Proceedings of the First International Conference on Agreement Technologies, AT 2012, Dubrovnik, Croatia, October 15-16, 2012*, pages 378–392, 2012.

Saul Kripke. Speaker's reference and semantic reference. *Midwest studies in philosophy*, 2(1):255–276, 1977.

Y. Labrou, T. Finin, and Yun Peng. Agent communication languages: the current landscape. *IEEE Intelligent Systems*, 14(2):45–52, mar 1999. doi: 10.1109/5254.757631.

Loredana Laera, Ian Blacoe, Valentina A. M. Tamma, Terry R. Payne, Jérôme Euzenat, and Trevor J. M. Bench-Capon. Argumentation over ontology correspondences in MAS. In *6th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2007), Honolulu, Hawaii, USA, May 14-18, 2007*, page 228, 2007. doi: 10.1145/1329125.1329400.

B. M. Lake, R. Salakhutdinov, and J. B. Tenenbaum. Human-level concept learning through probabilistic program induction. *Science*, 350(6266):1332–1338, dec 2015. doi: 10.1126/science.aab3050.

Mr John Leland. *Hip: the history*. Harper Collins, 2009.

David Lewis. *Convention: A Philosophical Study*. Harvard University Press, 1969.

Juanzi Li, Jie Tang, Yi Li, and Qiong Luo. Rimom: A dynamic multistrategy ontology alignment framework. *IEEE Trans. Knowl. Data Eng.*, 21(8):1218–1232, 2009. doi: 10.1109/TKDE.2008.202.

Yuhua Li, David McLean, Zuhair Bandar, James O'Shea, and Keeley A. Crockett. Sentence similarity based on semantic nets and corpus statistics. *IEEE Trans. Knowl. Data Eng.*, 18(8):1138–1150, 2006. doi: 10.1109/TKDE.2006.130.

Brian MacWhinney. *The CHILDES project: The database*, volume 2. Psychology Press, 2000.

Jonathan Malmaud, Earl Wagner, Nancy Chang, and Kevin Murphy. Cooking with semantics. In *Proceedings of the ACL 2014 Workshop on Semantic Parsing*. Association for Computational Linguistics, 2014. doi: 10.3115/v1/w14-2407.

Elisa Marengo, Matteo Baldoni, Cristina Baroglio, Amit K. Chopra, Viviana Patti, and Munindar P. Singh. Commitments with regulations: reasoning about safety and control in REGULA. In *10th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2011), Taipei, Taiwan, May 2-6, 2011, Volume 1-3*, pages 467–474, 2011.

Deborah L McGuinness, Frank Van Harmelen, et al. OWL web ontology language overview. *W3C recommendation*, 10(10):2004, 2004.

Fiona McNeill and Alan Bundy. Dynamic, automatic, first-order ontology repair by diagnosis of failed plan execution. *Int. J. Semantic Web Inf. Syst.*, 3(3):1–35, 2007. doi: 10.4018/jswis.2007070101.

Rada Mihalcea, Courtney Corley, and Carlo Strapparava. Corpus-based and knowledge-based measures of text semantic similarity. In *Proceedings, The Twenty-First National Conference on Artificial Intelligence and the Eighteenth Innovative Applications of Artificial Intelligence Conference, July 16-20, 2006, Boston, Massachusetts, USA*, pages 775–780, 2006.

Nicole Mills. Situated learning through social networking communities: The development of joint enterprise, mutual engagement, and a shared repertoire. *CALICO Journal*, 28(2):345–368, jan 2011. doi: 10.11139/cj.28.2.345-368.

Marco Montali. *Specification and Verification of Declarative Open Interaction Models - A Logic-Based Approach*, volume 56 of *Lecture Notes in Business Information Processing*. Springer, 2010. doi: 10.1007/978-3-642-14538-4.

David Nunan. *Task-Based Language Teaching*. Ernst Klett Sprachen, 2006.

Santiago Ontañón and Enric Plaza. Concept convergence in empirical domains. In *Discovery Science - 13th International Conference, DS 2010, Canberra, Australia, October 6-8, 2010. Proceedings*, pages 281–295, 2010. doi: 10.1007/978-3-642-16184-1_20.

Lluís Padró and Evgeny Stanilovsky. Freeling 3.0: Towards wider multilinguality. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation, LREC 2012, Istanbul, Turkey, May 23-25, 2012*, pages 2473–2479, 2012.

Paolo Pareti. Distributed linked data as a framework for human-machine collaboration. In *Proceedings of the 7th International Workshop on Consuming Linked Data co-located with 15th International Semantic Web Conference, COLD@ISWC 2015, Kobe, Japan, October 18, 2016.*, 2016.

Paolo Pareti. Code for protocol generators. `https://github.com/paolo7/protocol-generators`, 2017a. Accessed: 26/6/2017.

Paolo Pareti. Multilingual wikihow human instructions. `https://www.kaggle.com/paolop/human-instructions-multilingual-wikihow`, 2017b. Accessed: 26/6/2017.

Paolo Pareti. *Representation and Execution of Human Know-How on the Web*. PhD thesis, University of Edinburgh, School of Informatics, 2017c.

Paolo Pareti, Benoit Testu, Ryutaro Ichise, Ewan Klein, and Adam Barker. Integrating know-how into the linked data cloud. In *Knowledge Engineering and Knowledge Management - 19th International Conference, EKAW 2014, Linköping, Sweden, November 24-28, 2014. Proceedings*, pages 385–396, 2014. doi: 10.1007/978-3-319-13704-9_30.

Paolo Pareti, Ewan Klein, and Adam Barker. Linking data, services and human know-how. In *The Semantic Web. Latest Advances and New Domains - 13th International Conference, ESWC 2016, Heraklion, Crete, Greece, May 29 - June 2, 2016, Proceedings*, pages 505–520, 2016. doi: 10.1007/978-3-319-34129-3_31.

Rohit Parikh. Vagueness and utility: The semantics of common nouns. *Linguistics and Philosophy*, 17(6):521–535, dec 1994. doi: 10.1007/bf00985317.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1532–1543, 2014.

Maja Pesic and Wil M. P. van der Aalst. A declarative approach for flexible business processes management. In *Business Process Management Workshops, BPM 2006 International Workshops, BPD, BPI, ENEI, GPWW, DPM, semantics4ws, Vienna, Austria, September 4-7, 2006, Proceedings*, pages 169–180, 2006. doi: 10.1007/11837862_18.

Maja Pesic, Helen Schonenberg, and Wil M. P. van der Aalst. DECLARE: full support for loosely-structured processes. In *11th IEEE International Enterprise Distributed Object Computing Conference (EDOC 2007), 15-19 October 2007, Annapolis, Maryland, USA*, pages 287–300, 2007. doi: 10.1109/EDOC.2007.14.

Martin J. Pickering and Simon Garrod. Toward a mechanistic psychology of dialogue. *Behavioral and Brain Sciences*, 27(02), apr 2004. doi: 10.1017/s0140525x04000056.

Stefan Poslad, Phil Buckle, and Rob Hadingham. The fipa-os agent platform: Open source for open standards. In *proceedings of the 5th international conference and exhibition on the practical application of intelligent agents and multi-agents*, volume 355, page 368, 2000.

Willard V. O. Quine. *Word & Object*. MIT Press, 1960.

Willard V. O. Quine. Indeterminacy of translation again. *The Journal of Philosophy*, 84(1):5, jan 1987. doi: 10.2307/2027132.

Anand S. Rao and Michael P. Georgeff. BDI agents: From theory to practice. In *Proceedings of the First International Conference on Multiagent Systems, June 12-14, 1995, San Francisco, California, USA*, pages 312–319, 1995.

David Robertson. A lightweight coordination calculus for agent systems. In *Declarative Agent Languages and Technologies II, Second International Workshop, DALT 2004, New York, NY, USA, July 19, 2004, Revised Selected Papers*, pages 183–197, 2004. doi: 10.1007/11493402_11.

Bertrand Russell. On denoting. *Mind*, 14(56):479–493, 1905.

Gabrielle Santos, Valentina A. M. Tamma, Terry R. Payne, and Floriana Grasso. A dialogue protocol to support meaning negotiation.: (extended abstract). In *Proceedings of the 2016 International Conference on Autonomous Agents & Multiagent Systems, Singapore, May 9-13, 2016*, pages 1367–1368, 2016a.

Marc Ericson C. Santos, Arno in Wolde Lbke, Takafumi Taketomi, Goshiro Yamamoto, Ma. Mercedes T. Rodrigo, Christian Sandor, and Hirokazu Kato. Augmented reality as multimedia: the case for situated vocabulary learning. *Research and Practice in Technology Enhanced Learning*, 11(1), jan 2016b. doi: 10.1186/s41039-016-0028-2.

Pol Schumacher, Mirjam Minor, Kirstin Walter, and Ralph Bergmann. Extraction of procedural knowledge from the web: a comparison of two workflow extraction approaches. In *Proceedings of the 21st World Wide Web Conference, WWW 2012, Lyon, France, April 16-20, 2012 (Companion Volume)*, pages 739–747, 2012. doi: 10.1145/2187980.2188194.

John R Searle. *Speech acts: An essay in the philosophy of language*, volume 626. Cambridge university press, 1969.

Stuart C. Shapiro. *Encyclopedia of Artificial Intelligence*. John Wiley & Sons, Inc., New York, NY, USA, 2nd edition, 1992.

Nuno Silva, Paulo Maio, and João Rocha. An approach to ontology mapping negotiation. In *Integrating Ontologies '05, Proceedings of the K-CAP 2005 Workshop on Integrating Ontologies, Banff, Canada, October 2, 2005*, 2005.

M. P. Singh. Commitments in multiagent systems: Some history, some confusions, some controversies, some prospects. In Tummolini L. Falcone R. Miceli M. Paglieri, F., editor, *The goals of cognition: Essays in honor of Cristiano Castelfranchi*, pages 1–29. 2012.

Munindar P. Singh. An ontology for commitments in multiagent systems. *Artif. Intell. Law*, 7(1):97–113, 1999. doi: 10.1023/A:1008319631231.

Munindar P. Singh. A social semantics for agent communication languages. In *Issues in Agent Communication*, pages 31–45, 2000. doi: 10.1007/10722777_3.

Jeffrey Mark Siskind. A computational study of cross-situational techniques for learning word-to-meaning mappings. *Cognition*, 61(1-2):39–91, oct 1996. doi: 10.1016/s0010-0277(96)00728-7.

John F. Sowa. *Knowledge representation: logical, philosophical, and computational foundations*. Brooks/Cole, 2000.

Luc Steels. A self-organizing spatial vocabulary. *Artificial Life*, 2(3):319–332, 1995. doi: 10.1162/artl.1995.2.3.319.

Luc Steels. The origins of ontologies and communication conventions in multi-agent systems. *Autonomous Agents and Multi-Agent Systems*, 1(2):169–194, 1998. doi: 10.1023/A:1010002801935.

Angus Stevenson. *Oxford Dictionary of English*. Oxford University Press, 2010.

Richard S. Sutton and Andrew G. Barto. Reinforcement learning: An introduction. *IEEE Trans. Neural Networks*, 9(5):1054–1054, 1998. doi: 10.1109/TNN.1998.712192.

Moritz Tenorth, Daniel Nyga, and Michael Beetz. Understanding and executing instructions for everyday manipulation tasks from the world wide web. In *IEEE International Conference on Robotics and Automation, ICRA 2010, Anchorage, Alaska, USA, 3-7 May 2010*, pages 1486–1491, 2010. doi: 10.1109/ROBOT.2010.5509955.

Edward L Thorndike. Animal intelligence: an experimental study of the associative processes in animals. *The Psychological Review: Monograph Supplements*, 2(4):i, 1898.

Peter D. Turney and Patrick Pantel. From frequency to meaning: Vector space models of semantics. *J. Artif. Intell. Res.*, 37:141–188, 2010. doi: 10.1613/jair.2934.

Wil M. P. van der Aalst. *Process Mining - Discovery, Conformance and Enhancement of Business Processes*. Springer, 2011. doi: 10.1007/978-3-642-19345-3.

Wil M. P. van der Aalst and Maja Pesic. Specifying and monitoring service flows: Making web services process-aware. In *Test and Analysis of Web Services*, pages 11–55. 2007.

Jurriaan van Diggelen, Robbert-Jan Beun, Frank Dignum, Rogier M. van Eijk, and John-Jules Ch. Meyer. ANEMONE: an effective minimal ontology negotiation environment. In *5th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2006), Hakodate, Japan, May 8-12, 2006*, pages 899–906, 2006. doi: 10.1145/1160633.1160794.

Jurriaan van Diggelen, Robbert-Jan Beun, Frank Dignum, Rogier M. van Eijk, and John-Jules Ch. Meyer. Ontology negotiation in heterogeneous multi-agent systems: The ANEMONE system. *Applied Ontology*, 2(3-4):267–303, 2007.

Rogier M. van Eijk, Frank S. de Boer, Wiebe van der Hoek, and John-Jules Ch. Meyer. On dynamically generated ontology translators in agent communication. *Int. J. Intell. Syst.*, 16(5):587–607, 2001. doi: 10.1002/int.1025.

Willem Robert van Hage, Hap Kolb, and Guus Schreiber. Relevance-based evaluation of alignment approaches: The OAEI 2007 food task revisited. In *Proceedings of the 3rd International Workshop on Ontology Matching (OM-2008) Collocated with the 7th International Semantic Web Conference (ISWC-2008), Karlsruhe, Germany, October 26, 2008*, 2008.

Mahadevan Venkatraman and Munindar P. Singh. Verifying compliance with commitment protocols. *Autonomous Agents and Multi-Agent Systems*, 2(3): 217–236, 1999. doi: 10.1023/A:1010056221226.

Ludwig Wittgenstein. *Philosophical Investigations*. Basil Blackwell, Oxford, 1953.

Fei Xu and Joshua B Tenenbaum. Word learning as bayesian inference. *Psychological review*, 114(2):245, 2007.

Chen Yu and Dana H. Ballard. A unified model of early word learning: Integrating statistical and social cues. *Neurocomputing*, 70(13-15):2149–2165, 2007. doi: 10.1016/j.neucom.2006.01.034.

Benjamin D. Zinszer, Sebi V. Rolotti, Fan Li, and Ping Li. Bayesian word learning in multiple language environments. *Cognitive Science*, nov 2017. doi: 10.1111/cogs.12567.

Appendix A

References to Software

In Chapters 3, 4 and 5 we evaluated our techniques using simulations over data generated by ourselves. The implementation of these simulations is publicly available in online repositories. In this way we hope to let readers reproduce the experiments, as well as reuse the software for other purposes. In this appendix we provide links to the repositories as well as a short explanation about their content. A detailed explanation about the software can be found in the README page of each repository.

All the repositories can be found in the GitHub account of the author of this dissertation (`https://github.com/paulachocron`). In all cases, the URLs that we provide here were last accessed on 13 March 2018. We do not provide the code or data we used in Chapter 6. We made this decision for two reasons. First, this work was developed jointly with an external collaborator. Second, the results were not yet published by the date of dissemination of the dissertation. The software corresponding to this chapter may appear later under the same GitHub account.

We run all these experiments in a server with an `Intel Xeon E3-1246 v3` CPU. When possible, we include the datasets that we used to make the experiments completely reproducible. However, it is necessary to take into account that all these experiments have aleatory aspects so, while the expected results after many experiments should be similar to the ones we show, they can vary for individual runs.

A.1 Software for Chapter 3

The software for the main evaluation of Chapter 3 is the repository `alignment-learning-open` (which can be found in `https://github.com/paulachocron/\learning-alignment-fsm.git`).

We provide code for the generation of finite state machines representing pro-

tocols and alignments as well as the implementation of agents that infer alignments with the four proposed approaches. We also include the implementation of the simulation that makes agents interact measuring their successes. The code allows to run the two experiments presented in Section 3.6. We include a demo with detailed instructions to run it, that shows the behavior of agents in a comprehensive way.

A.2 Software for Chapter 4

The software for the main evaluation of Chapter 4 is the repository `alignment-learning-open` (which can be found in `https://github.com/paulachocron/\alignment-learning-open.git`).

The repository includes code to generate ConDec protocols, as well as a implementation of its interface with the NuSMV model checker that allows to decide satisfiability. We provide the implementation of agents 'simple', 'reasoner', 'student', 'teacher', and 'logical'. The code allows to run the four experiments presented in Section 4.5, as well the ones corresponding to the p-necessary constraints and the deduction-based approach. Again, we provide a demo with detailed instructions to run it.

A.3 Software for Chapter 5

The software for the main evaluation of Chapter 5 is the repository `commitment-semantics-learning` (which can be found in `https://github.com/paula\chocron/commitment-semantics-learning.git`).

The repository includes the code to generate commitment specifications, as well as the different implementations of the student, taking into account the three extensions that we considered. The code allows to execute the experiments 1 and 2, that measure the amount of successful interactions and the precision with respect to a reference alignment, respectively. A detailed explanation of how to run these experiments and which arguments they accept can be found in the README file.