Master of Science in Informatics at Grenoble
Master Mathématiques Informatique - spécialité Informatique
option Artificial Intelligence and the Web

Ontology Evolution Through Interaction

Irina Teodora Dragoste

2 September 2016

Research project performed at INRIA

Under the supervision of:
Dr. Jérôme Euzenat, INRIA

Dr. Manuel Atencia, Univ Grenoble Alpes

Defended before a jury composed of:
Prof. James L. Crowley
Prof. Massih-Reza Amini
Dr. Hubert Garavel
Dr. Jérôme Euzenat
Dr. Carole Adam

September 2016

This work has been partially supported by the LabEx PERSYVAL-Lab (ANR-11-LABX-0025-01) funded by the French program Investissement d'avenir.

Abstract

This paper shows how agents can evolve their ontologies while trying to communicate, following a protocol inspired by cultural evolution experiment methodology. While other studies have dealt with evolving ontology alignments, we are interested in refining the classification precision, which is accomplished during the protocol: while communicating about objects encountered in the environment, agents may discover the need to increase their classification precision in order to reach an agreement with the interlocutor. They do so by creating and adopting new categories from other agents that would help them discriminate better. We conducted several experiments to show that a population playing this game will always converge to a state of full communication success, when agents have stabilized their knowledge and cannot learn any new categories.

Contents

— 1 —
Introduction

The **goal** of our research is to show that it is possible for agents to refine their knowledge while trying to **improve communication**. Representing knowledge as ontologies, we define **ontology refinement** as improving the way agents classify observed objects from the environment to obtain more precision.

1.1 Problem statement

For the purpose of reaching our goal, we need to answer the following research questions:

- Can we define an experimental set-up for ontology evolution?
- How is evolution influenced by the initial situation:
  - the prior knowledge of agents
  - the environment complexity, in terms of features and feature values objects from the environment may have

A possible approach for investigating how agents could evolve their ontologies is simulating the process of **cultural evolution**.

1.2 Cultural Games for Enhancing Agent Ontologies by Acquiring Categories

Our approach to the problem stated above is to set up experiments that follow the approach of cultural evolution studies, that would help us answer our research questions.

This involves designing a communication protocol, called *game*, in which randomly selected agents interact with each other, trying to communicate about objects from the environment. Along these repeated interactions, the evolution of some parameters is monitored, among which communication failure/success rate. The outcome of each interaction is the status of local communication, which can be failure or success. The goal of agents is to minimize the global communication failure rate, by performing a certain repair operation when local failure occurs.

The parameters that we need to define for instantiating a cultural evolution game with the scope of refining ontology content are **communication failure** and **repair operation**.

We consider that agent ontologies contain categories of interest, into which agents classify objects based on their observable features. Depending on their own experience and prior knowledge, agents in the same environment can have different categories of interest. These categories may have unobservable properties (for example, poisonous mushrooms) that agents need to learn from one another.

Thus, in the game we design, we consider communication to be **unsuccessful** when one agent observes that the other has a more discriminating classification. The **repair operation** is defined as adopting the classes with assumed greater classification precision.

The expected result of a population playing this game is enhancing agent ontologies by enriching their categories, depending on their prior knowledge.

Exemplified Ontology Enhancement Game

Pascal and Mary both observe a white square. Pascal identifies the object as *Blanc*, based on its colour. Mary finds that the object belongs to both the categories of *White* and *Square* objects. Therefore, she creates a new category, *WhiteSquare*, representing those objects that have a white colour and a square shape. Mary tells Pascal that the object is a *WhiteSquare* and Pascal records that there should be a relation between what he calls *Blanc* and what Mary calls *WhiteSquare*. Since he had no previous relations between Blanc and Mary's terminology, he simply records it (we call this a **success**).

If Pascal would remember a previous interaction with Mary in which she called *White* another object that Pascal called *Blanc*, Pascal would understand that Mary makes a difference between *White* and *WhiteSquare* that he does not make. We call this a **failure** (because Pascal realises that he fails to understand precisely Mary). In order to recover from this failure, he would ask what difference she makes between *White* and *WhiteSquare*. Mary answers that *White* objects are those whose colour is white and *WhiteSquare* objects are those who, in addition, have a square shape. Because they both share the feature vocabulary, Pascal can understand these definitions and integrate the in his own classification. Pascal's *Blanc* is the same as Mary's *White*, but he can create a new subcategory *WhiteSquare_from_Mary*.

We can imagine with this short example, how agents sharing features for describing the world may evolve their ontologies through **discovering the world**(Mary creating *WhiteSquare*) or **communicating** (Pascal learning it).

1.3 Outline

The methodology we are about to follow in our research consists in the steps below:

1. study related work on **cultural knowledge evolution** and **artificial agents simulations** for a deeper understanding of the problem and for situating our solution in the context of existing approaches

2. design an experiment consisting in a protocol of agent interaction that would allow a population of agents to evolve their ontologies while communicating, for any random initial conditions.

3. implement the experiment and integrate it in the existing framework of ontology network evolution experiments ([1])

4. formulate and test hypotheses, run the experiment, observe and analyse the results,

The outline of this report is organized with respect to the presented methodology. Thus Chapter 2 presents the context of our research, and state of the art cultural evolution experiments. Chapter 3 details the proposed cultural game, formalizes agent knowledge and environment representation and exemplifies the protocol on a few cases of interest. In Chapter 4 we present the results of several experiments, formulate hypotheses based on the observations and test them. Finally, Chapter 5 contains our concluding remarks regarding the current contributions and possible directions of improvement.

— 2 —
Related Work

2.1 Cultural Evolution

Cultural evolution studies investigate the emergence and continuous evolution of a shared cultural artifact in a population. Experiments with decentralized artificial agent systems have been used to simulate this phenomenon.

2.2 Cultural Evolution of Language

The collection of articles in [6] defends the *selectionist* theory of language evolution, that states that a shared language system can originate and evolve in a population by *recruitment* of grounded cognitive capabilities to generate conceptualization strategies, *selection* of strategies and language elements that improve communication and *self-organization* of agents by aligning their local solutions [7].

Experiments in [6] use a methodology for simulating the cultural process of evolution in a population of artificial agents through a series of local, decentralized interactions, called **language games**. They involve a hearer and a speaker trying to communicate about objects observed in the environment and updating their internal knowledge in order to succeed.

The cultural evolution terminology presented here will be detailed in the next section, exemplifying how language games can be used to study spatial language.

2.2.1 Evolving Grounded Spatial Language Strategies

While some experiments in [6] only explore the emergence of vocabularies, others show their co-evolution with and influence by grammatical structures. Following the latter, [4] experimentally explores and demonstrates the selectionist theory of spatial language evolution using of a *whole system approach* that has been validated by successfully reconstructing German locative grammar and has shown that the use of grammar improves communication success.

Syntax influences the semantics of phrases and encodes a certain internal representation of reality, called **conceptualization strategy**. A conceptualization strategy can be formalized as a composition of cognitive operations an agent can execute to identify an object in space, for which and implementation called *Incremental Recruitment Language* is proposed. In spatial languages, multiple conceptualization strategies can be used to represent the same spatial reality, varying in spatial relations systems (for example, proximal, projective or absolute), perspectives, landmarks and frames of reference.

The article [5] explores the origin and evolution of spatial conceptualization strategies and their influence on spatial vocabulary through various experiments that respect the **language games** methodology from [6]. It demonstrates that through communication, a population that initially shares only cognitive capabilities and has no local conceptualization strategies, spatial concepts or lexemes can emerge to a shared spatial language system.

Spatial language game

The adapted **language game** is a situated experiment, with humanoid robots observing various types object in a spatial environment. Each iteration involves selecting randomly a *speaker* and *hearer* from the population, as well as an object that will be the communication *topic*: i) the speaker tries to find a conceptualization strategy to describe the topic; ii) the speaker translates the conceptualization into language, based on its private vocabulary; iii) the speaker utters the production to the hearer; iv) the hearer tries to parse it, using its own strategies and concept names; v) the hearer uses the decoded conceptualization strategy to identify the assumed topic and indicates it non-verbally; vi) if it coincides with the intended topic, the game is *successful*. Otherwise, *communication failure* is signaled by the speaker, which points to the intended topic.

Failure can occur at various points in the protocol, and agent can use various *repair operations* (*invention*, *adoption*, *alignment*) to improve the success rate. They always select the most distinctive and heuristically successful conceptualization strategy, spatial category or lexeme when different alternatives are available. The repair operations depend on the scope of the experiment, as detailed in the three experiments of increasing complexity below.

Evolution of Grounded Lexicons

Grounded on a preexisting conceptualization strategy, agents *invent* (failure at step i.) and *adopt* (failure at step iv.) spatial categories and corresponding lexicons, *align* categories in successful interactions by updating their sample set, and *align* lexicons heuristically, by rewarding successful category-lexicon constructions and punishing unsuccessful ones.

Thus, a population with empty initial spatial ontologies and lexicons can form a language system with increasingly similar categories, that tends to full communication success.

Selection and Alignment of Spatial Conceptualization Strategies

In this experiment, the initial state contains **different private conceptualization strategies**. Using the same **repair operations** as in the previous experiment, competing strategies and their corresponding language systems are selected and aligned by rewarding or punishing them, depending on communication success.

The preference of a strategy over another is highly influenced by the environment, and it also encourages inventing categories of the preferred system while inhibiting the others. The result is a single surviving language system.

Recruitment of Conceptualization Strategies

In addition to the previous operations, agents can *invent* spatial conceptualization strategies are *invented* through *recruitment* (i.e. combination of grounded spatial cognitive operations). This allows more complex **repair** solutions:

- when the speaker fails to conceptualize (at step i.), it invents new strategies and associated categories, it also invents new categories for existing strategies, and then selects the most *discriminating* solution, for which it invents the corresponding lexeme.

- when the hearer fails, it invents new strategies and a new category corresponding to the pointed topic, for each strategy. The most *discriminating* strategy and category pair is selected, and a corresponding lexeme is invented.

In a population that initially shares only cognitive functions, agents are able to generate different conceptualization strategies and associated language systems from scratch, converging to a single strategy and system in the end, with increasing success rate and concept similarity.

2.2.2 Cultural Evolution of Knowledge Representation - Ontology Alignment Repair

The article [1] represents a first attempt of applying cultural evolution to **knowledge representation**. Here, knowledge is represented as ontologies, the artefact under evolution being **ontology alignments**.

Overview

An **alignment** is a set of correspondences between elements (classes or properties) belonging to two different ontologies. As explained in [2], a correspondence $< e1, e2, r >$ consist of a relation r (equivalence, disjointness, more general or less general) between an entity e_1 from the first ontology and an entity e_2 from the second one. A set of ontologies and their alignments creates an **ontology network**. A network is symmetric when, for every alignment between an ontology o_1 and another o_2, there is also its converse alignment between o_2 and o_1 [1]. An alignment can be incoherent [3] when there exists an unsatisfiable concept in the merged ontology (resulted from two ontologies and their alignment) that is satisfiable in one of the ontologies.

The scope of this paper is restricted to improving incorrect alignments (alignment repair), and its aim is to show experimentally that through a simplified cultural evolution protocol called **cultural alignment repair**, agents will always reach a state characterized by complete coherence and correctness of alignments, successfully communication and improved alignments quality.

Cultural repair is characterized by agents attempting to communicate and overcoming failure through repair actions, that would improve communication. Agents do not change their state as long as communication is successful, and the side effect of their repeated local repair action is the evolution of the entire population.

Adapted to a network of consistent ontologies, each agent has access only to its own ontology and his interfaces (alignments) to other agents. An alignment models an agent's assumptions about the other's knowledge. All agents trust their interlocutors and initially assume their correspondences to be correct. When, through communication, an agent encounters an inconsistency, it repairs it by reducing the correspondence scope.

Experimental setup

The proposed cultural repair game is intended to be very simple; being a cultural evolution experiment, it is characterized by a communication protocol between agents, that is repeated in iterations with randomized parameters (object of communication, interlocutors), thus simulating complete decentralization.

The **environment** contains equiprobable objects with n ordered binary features, and n equiprobable agents, each having a different ontology. The **ontologies** used in this experiment are simple taxonomies with disjoint relationships, in the form of complete decision trees where each level corresponds to a feature, using $k-1$ features. The root of all decision trees is a special category that classifies all objects in the environment. Considering the features in a circular permutation, the k-th agent uses no feature at the first level, the k-th feature at the second level, the $k+1$-th at the third and so on until the $k+n-2$-th at the last level. The ontology network is **initialized** with a symmetric alignment, containing equivalence between the top concepts, and a set of randomly generated equivalence correspondences between classes. This may contain incorrect correspondences that would be repaired during the experiment.

The game **protocol** states that in each iteration, two randomly selected agents classify a randomly generated object to its most specific class. The first agent uses its alignment to the second agent to compare the results. If they are inconsistent (in the first ontology, the result (*target*) is disjoint from the *source* class that corresponds to the second agent's result), this is considered a **failure**, and the first agent takes a **repair** action to weaken the detected incorrect correspondence.

Results and Analysis

Experiments have been run by playing the game with $n \leq 4$ (for non-trivial results) in order to prove the hypothesis and analyse the resulting state of the ontology network. The quality of this state is analysed in terms of *average degree of incoherence* and *semantic F-measure* with respect to the reference alignments.

By monitoring the rate of successful iterations, experiments show that after a certain number of iterations in which incorrect correspondences are repaired, the **success rate** always converges asymptotically to 1, regardless of the game randomness (initial alignment, the object and interlocutors of each round).

Different repair strategies that are entailed by the initial correspondence, reduce its scope and avoid the current communication failure can be used: **delete** simply removes an incorrect correspondence, *replace* weakens it from *equivalent* to *less general*, and *add* which, in addition to **replace**, creates a *more general than* correspondence from the parent of the *target* to the *source*. All the repair strategies eventually result in a fully coherent network and suppress all incorrect correspondences. The more the correspondence is weakened, the faster the game converges to full success, but the lower the final recall is. Thus, although the slowest, **add** strategy is the most efficient and improves the initial F-measure.

The quality of the cultural repair with **add** method is compared to the results obtained using **alignment repair algorithms** *Alcomo* (which discards correspondences) and *LogMap* (which weakens them). These two algorithms may preserve incorrect correspondences as long as the network is coherent. This may lead to less syntactic precision, but more semantic recall than cultural repair. There is a large variation of F-measure results due to the dependence on the initial alignment, but generally all three methods slightly improve the initial F-measure and obtain comparable values. It was experimentally observed that by increasing the size of the problem ($n > 4$), the cultural method improves and obtains better results than the repair algorithms. It is shown that the cultural repair results improve and **exceed** the other methods by increasing the number of **initial correspondences**. This is explained by the tendency of the repair algorithms to retain incorrect correspondences.

Conclusion

The article demonstrates that a very simple cultural evolution game can be used to successfully repair alignments in a network, with agents acting locally and without knowledge of alignment coherence. The resulting convergence state is fully coherent, is followed by successful communication, and its quality is on par with well-known alignment repair algorithms. In contrast with these algorithms, cultural repair never ignores incorrect correspondences, which can lead to a higher F-measure quality of the result for a large initial alignment.

As agents use alignments to communicate, it is shown that they do not require correct initial alignments to interact, are able to overcome communication failure locally and globally repair the network alignment.

2.3 Conclusion

We have presented how cultural evolution games can successfully be used in experiments in order to study the evolution of complex knowledge, like natural language [6]. The experiments can lead to impressive results, allowing to study and reconstruct natural language from conceptualizing about spatial relations, to grammar and vocabulary, starting only from shared cognitive capabilities in a population [5].

Our proposed ontology evolution game follows [1]'s direction of studying cultural evolution of knowledge representation in the form of ontologies. [1] is not concerned with evolving the content of ontologies themselves, but with the alignment between them that facilitate communication. We are interested in studying how interactions between agents can influence the content of the ontology, so instead of simply repairing alignment by discarding correspondences, we will try to learn how to refine ontologies.

— 3 —

Proposed Ontology Evolution Game

3.1 Ontology Evolution Game

In order to reach our goal of investigating if it is possible for agents to refine their ontologies while communicating, we have designed and implemented a cultural evolution experiment, in the context of an existing experimental framework [1] for cultural evolution of alignments in a network of ontologies.

The main characteristic of the experiment is that agents share the same conceptual framework (consisting in features) on the base of which they construct different ontologies.

The experiment setup is composed of the following elements:

Environment consisting of objects described by the complete set of n features. Each feature F_i can have m_i possible **distinct** feature values, $m_i \geq 2$. The objects are equally distributed with respect to the feature values.

Population k agents $A_1 \ldots A_k$ with corresponding ontologies $\Theta_1 \ldots \Theta_k$. The ontologies contain:

- **shared elements**, with an implicit alignment between them:
  - category *Object* classifying all objects in the environment
  - all the defined features as object properties
  - all their corresponding property values
- **private elements**: A randomly generated set of classes called *categories*, used to classify objects from the environment based on their features. They are subclasses of *Object*, formed as a conjunction between 0 to n constraints on property values. All the possible categories are uniformly distributed.

Agents record their interactions with other agents in their private memory.

Game A game is composed of several rounds. In each round, a pair of agents is selected at random with equal probability, and an object is generated at random from the environment. Both the agents classify the object into the most specific category of their ontology. If the object belongs to more than one most specific category, the intersection class between them is created and added to the ontology. The second agent tells the first agent his classification result, and the first agent records it into his memory, along with the class to which the object belongs in his own memory.

Failure The first agent can find in memory previous interactions where he used the same category as in the current round, while the second agent used different categories. This indicates that the first agent fails to distinguish between objects in a case when the second agent is able to, and the second agent has a finer classification than the first, so the round it is considered a **failure**. Otherwise, the round is considered successful.

Repair is the action performed by the first agent in the case of failure. He asks the second agent for the definitions of the two different categories, and adopts them into his ontology. In order to avoid synonymy, a category is not adopted if the ontology already contains an equivalent class.

Success measure is defined as the number of successful rounds divided by the total number of rounds in a game.

Other measurements we also compute the number of new classes learned by each agent, either by intersection with its own existing categories, or by adoption from other agents.

Thus, by following the simple communication protocol, there are two ways an agent can obtain new *categories*:

- by **intersection**, inventing new categories on its own, based on existing ones, while discovering the environment.
- by **adoption**, acquiring categories from other agents, through communication

While through intersection, an agent always creates a more precise category than his current classification result, this is not necessarily true with categories obtained through adoption, which can be less precise or at the same precision level. Nevertheless, adopting these categories helps the agent discriminate between objects it could not before, and they can help create further more categories through intersection. We will later show experimentally that adopting more general or just-as-general categories, after several iterations, will contribute to increasing classification precision overall in the population.

3.2 Formal Description

This section formalizes the concepts involved in the experiment (environment, agent, object) and details how agents can represent and operate with knowledge.

3.2.1 Representing Agent Knowledge

Agent ontologies represent their knowledge about the environment. We have chosen *Description Logics* to formalize these ontologies.

Shared Knowledge About The Environment

Let there be an environment containing the set of n features f_1 ... f_n. All agents contain the following set of axioms:

$$Object \equiv \exists f_1.\top \sqcap \ldots \sqcap \exists f_n.\top \tag{3.1}$$

which describe the class of all objects in the given environment, and state that *all* objects in the environment must have all the established features.

For each feature f_i with corresponding feature values $F_i = \{f_i^j | j = \overline{2...m_i}\}$, all agents have an axiom

$$Object \sqsubseteq \forall f_i.\{f_i^1, \ ... \ f_i^{m_k}\} \tag{3.2}$$

which restricts the possible values of property f_i that objects can have in the environment to elements of the set F_i

f_i is functional
values $\{f_i^j | j = \overline{2...m_i}\}$ are all distinct

We establish through a convention that all agents

- share the same knowledge about features, represented into the axioms above (contain *Object class*, the same properties and properties values, with the same corresponding names)

- are aware that they share this knowledge

Thus, when an agent receives a message containing statements about properties or property values from another agent, it knows that they are equivalent to the properties or property values with the same name from its own ontology.

Agent Private Categories

Agent private *category* definitions can be generated from axiom 3.1, instantiating the top concept $\top$ by replacing it with a feature value in none, some or all feature restrictions. For example, a category of agent A_k that constraints feature f_1 to value f_1^a, and feature f_3 to value f_3^b is defined as

$$A_k : f_1^a f_3^b \equiv \ \exists \ f_1.\{f_1^a\} \sqcap \exists \ f_2.\top \sqcap \exists \ f_3.\{f_3^b\} \sqcap ... \sqcap \exists \ f_n.\top \tag{3.3}$$

which entails that category $A_k : \ f_1^a \ f_3^b$ is a subclass of *Object* ($A_k : \ f_1^a \ f_3^b \sqsubseteq Object$). In order to avoid synonymy, no two equivalent categories will exist in an agent's ontology. This leads to a total number of $(|F_1|+1) \times ... \times (|F_n|+1)$ possible *categories* an agent can have, including *Object* category.

We must mention that axioms 3.3 are equivalent to conjunctions of restrictions on property values. We have chosen to represent only simple ontologies with conjunctions for this phase of our experiments, thus we do not include negation, or expressions like $f_1.\{f_1^a, f_1^b\} \sqcap ...$, which would introduce disjunction. Introducing disjunction and negation would significantly increase the number of possible categories.

This method leads to generating consistent, satisfiable ontologies, with categories defined as conjunctions of constraints on property values.

An agent A_i can only see his own ontology Θ_i, his own memory, or **classification history** T_i and the objects described by feature values.

3.2.2 Memory

The **classification history** table T_i keeps track of objects categorized by A_i and all the other agents it interacted with. $\forall$ agent i, $\exists$ table T_i:

Object	A_1	...	A_i	...	A_k
$F(o_j) =< F_1(o_j), ..., F_n(o_j) >$	$category(A_1, o_j)$	...	$category(A_i, o_j)$	...	$category(A_k, o_j)$

where $category(A_p, o_j)$ $\forall p$ is the signatures under which agent A_p has categorized object o_j. Because an agent only interacts with another agent during a round, say agent A_i only the value of $category(A_i, o_j)$ would be non empty on that row.

3.2.3 Communication Protocol

What agents can communicate to each other is:

- category signatures (random names associated to classes, depending on the agent) - for example *Hans:SchwartzeDreieck*
- category definitions (axiom defining a class as a conjunction of restrictions on certain properties) - for example $Hans{:}SchwartzeDreieck \equiv \exists color.\{black\} \sqcap \exists shape.\{triangle\}$

A **game round** is composed of the following steps:

1. Two agents A_i and A_j are selected at random from the existing k agents.
2. An object from the environment is generated at random. The object is completely described as a tuple of feature values for all its feature properties.
3. Agent i categorizes the generated object. If the object belongs to more than one "leaf" class in A_i's ontology then A_i's ontology is updated with their intersection class
4. A new line in table T_i is added for the generated object, and the label corresponding to its most specific class from A_i's ontology ($category(A_i, o)$), in A_i column of the table.
5. Agent A_i asks agent A_j to categorize the generated object. If the object belongs to more than one "leaf" class in A_j's ontology then A_j's ontology is updated with their intersection class
6. Agent A_j communicates agent A_i the signature of $category(A_j, o)$, corresponding to the most specific class object o belongs to in A_j's ontology. A_i stores this value in its T_i table at the new object's line, column A_j.
7. If agent A_i observes that agent A_j has a finer grain classification, discriminating between objects that A_i considers to be in the same category (i.e. $\exists$ o_p s.t. in classification history table T_i, $category(A_i, o) \equiv category(A_i, o_p)$ and $category(A_j, o) \neq category(A_j, o_p)$), then agent A_i asks for agent A_j's ontology class definition corresponding to the label $category(A_j, o_p)$ of the most recent such object o_p, as well as the definition of $category(A_j, o)$. If at least one of these classes did not already exist in A_i's ontology, it is **adopted** by A_i and the round status is considered **failure**. Otherwise, the round is **successful**.

3.2.4 Example

We provide an example by running the short example presented in the Introduction.

Game input

feature space: $color = \{black, white\}$ $shape = \{square, triangle\}$
agent ontologies:

<table>
<tr><th></th><th>Pascal's ontology</th><th>Mary's ontology</th></tr>
<tr><td>shared</td><td colspan="2">$Object \equiv \exists\, color.\top \sqcap \exists\, shape.\top$
$Object \sqsubseteq \forall\, color.\{black,\ white\}$
$Object \sqsubseteq \forall\, shape.\{square,\ triangle\}$
$color$ is functional
$shape$ is functional
$black \sqcap white \sqsubseteq \bot$
$square \sqcap triangle \sqsubseteq \bot$</td></tr>
<tr><td>private</td><td>Pascal:Blanc $\equiv \exists color.\{white\} \sqcap shape.\top$</td><td>Mary:White $\equiv \exists color.\{white\} \sqcap shape.\top$
Mary:Square $\equiv \exists color.\top \sqcap shape.\{square\}$</td></tr>
</table>

The agent's categories could be represented graphically:

Round 1

round 1: object=<white, triangle>, Pascal's turn

Pascal's classification history			
no.	object	*Pascal*	*Mary*
1	<white, triangle>	Pascal:Blanc	Mary:White

The result of this round is **success**, and agent categories remain unmodified

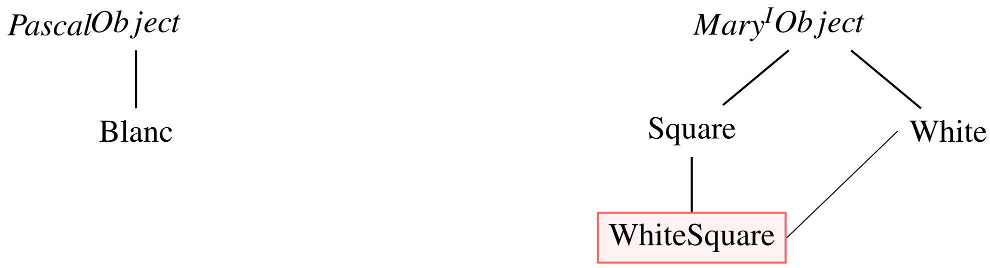
Round 2

round 2. <white, square>, Pascal's turn

Pascal's classification history			
no.	object	*Pascal*	*Mary*
1	<white, triangle>	**Pascal:Blanc**	Mary:White
2	<white, square>	**Pascal:Blanc**	Mary:WhiteSquare

As we can see, *Mary* applies **intersection** to learn a new category: *Mary:WhiteSquare*. It added the following axiom to its ontology ($Mary^I$):

$$Mary{:}WhiteSquare \equiv Mary{:}Square \sqcap Mary{:}White$$

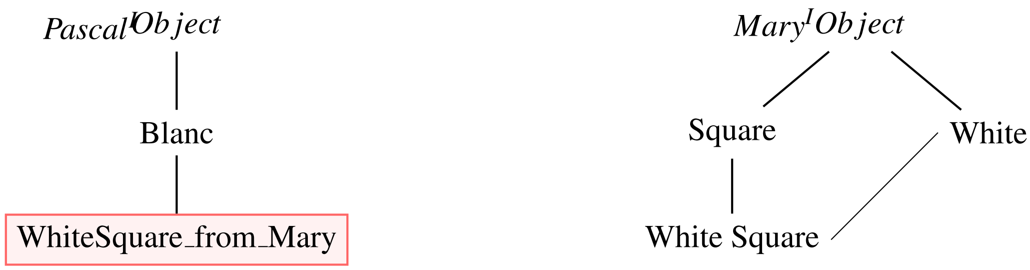


Pascal finds an entry in history where, for his same result *Pascal:Blanc*, Mary's result was *Mary:White*, different than her current result *Mary:WhiteSquare*. He applies **adoption** to learn to discriminate from Mary. Mary communicates him the following axioms:

$$Mary{:}WhiteSquare \equiv (\exists color.\top \sqcap shape.\{square\}) \sqcap (\exists color.\{white\} \sqcap shape.\top)$$
$$Mary{:}White \equiv \exists color.\{white\} \sqcap shape.\top$$

Because Pascal already has category *Pascal:Blanc* equivalent to *Mary:White*, it only adds the following axiom to his ontology ($Pascal^I$):

$$Pascal{:}WhiteSquare\_from\_Mary$$
$$\equiv (\exists color.\top \sqcap shape.\{square\}) \sqcap (\exists color.\{white\} \sqcap shape.\top)$$



The result of this round is **failure** (because at least one category was **adopted**). Both agents modify their categories by **intersection** and **adoption**.

3.3 Comparison With Related Work

What all agents share in both [1] and [5] is their ability to observe the same features of objects in the environment. In addition, our solution needed to represent this knowledge about features and their values in the agent's ontologies, and establish a convention to use identical names for them. Thus, agents in our proposed game also share a standard vocabulary about features, and, like in [1], about the most generic class. This was necessary because our solution involves agents communicating about features, shared knowledge about them being the building blocks of private knowledge.

In the cultural alignment repair experiment [1], agents represent their knowledge about each other through ontology alignments and perform repair operations to improve this knowledge. In our proposed game, agents assume correspondences between classification results, that are considered valid as long as both agents are able to discriminate between objects. The adopt operations do not repair these assumed correspondences directly, but bring agents closer to an agreement.

We propose this game as a **model of knowledge transmission** in a population of agents. Sharing a conceptual framework based on features and feature values, each agent can have a different classification system (here, ontology) composed of categories that the agent considers important based on some prior knowledge. Agents can evolve this classification system through experience i) by exploring objects in the environment and discovering new categories based on existing ones (when objects belong to more than one category, the agent is forced by the protocol to use **invention** in order to communicate a single classification result); ii) by communicating with other agents and learning from their experience (**adoption**).

3.4 Conclusion

We designed a cultural game for enhancing agent ontologies, where **culture** consists of categories of an ontology, that can be shared and evolve based on pre-existing shared properties. The initial ontologies are **grounded** on property names, and property value names. The properties represent observable features of objects that help distinguishing between them, and their grounding is based on the shared ability of agents to observe those feature values, and the convention of using the same names. Concepts are formed as constraints on property values, they are created by agents while observing objects from the environment based on already existing concepts, and shared between agents. Thus, agents evolve their initial ontologies (in terms of new concepts) while interacting.

— 4 —

Experiments

We conduct several experiments to show how the proposed game can be used for refining agent ontologies trough communication.

4.1 Experiment Goals

The goal of this study is to experimentally explore how agents can evolve their ontologies applying the protocol we defined in the previous chapter, according to cultural evolution experiments methodology.

- We wish to demonstrate that this simple cultural evolution protocol can be successfully adapted to explore how knowledge can evolve inside a population. We want to establish if, through the defined operations through which agents alter their knowledge to reach an agreement, ontologies always evolve. We wish to study the effect of sharing and combining knowledge about classification through a population (agents learn new classes from one another and combine them to create and transmit even more classes).

- We hypothesize that the population converges to a state with perpetually successful communication regardless of the initial conditions of the experiment and the randomness of each iteration.

- We want to explore the properties of this convergence state; Do the agents reach the same ontologies in the end? Do they reach the same level of precision, or ability to discriminate between objects?

- We also intend to discover conditions that influence ontology evolution. How is the output influenced by the initial agent knowledge? Does the randomness of each interaction influence the final output?

- We wish to study the scalability of our proposed game, exploring how the results are influenced by the environment complexity (in terms of feature space size) and by population size?

The next session presents the designed experimental settings required for studying the above stated concerns.

4.2 Experimental Settings

The **parameters** of the evolution game defined above are:

- the number of agents in a population
- the number of features defined in the environment
- the number of feature values defined for each feature

We have chosen to generate the initial ontologies of the evolution game randomly, based on the defined features. We have selected these parameters because they are easy to control and allow us to observe the general behaviour, regardless of the initial knowledge by running the game multiple times with different initial knowledge each time, and averaging over the results. Other parameters could be the *heterogeneity* of the initial ontologies, size of initial ontologies (number of categories), average degree of precision (generality/concreteness), number of feature values used in classification etc.

Based on these parameters, the initial settings of every game, as well as the settings of every round in a game, is **randomized**: At the beginning of a game, ontologies are initialized from a randomly generated subset of all the possible categories, where the probability of having each category is equally distributed. In order to have incomplete initial ontologies, we have established that the maximum size of this subset is $\prod_{i=1}^{n} |F_i|$ (the product of the sizes of all feature values for all features), and the minimum subset contains only *Object* category.

Algorithm 1 Initialize ontologies randomly

```
var initial_categories_limit = size(F1)*...*size(F2)
for  for i = 1, i++, while i < initial_categories_limit do
    Create category C
    Initialize set of restrictions for C with category Object
    for  each feature F_k do
        randomly decide to use F_k or not
        if use F_k then
            randomly select F_k feature's value F_k=f_k^m
            add F_k=f_k^m to the set of restrictions on C
        end if
        define C as the conjunction of the restrictions set
    end for
end for
```

- In each game round, the two distinct interacting agents are randomly selected, each agent in the population having the same probability of being selected first or second.
- The topic object of each round is randomly generated from the environment, with each feature value being equiprobable.

The game can consist in any number of rounds. We are interested in playing as many game rounds as necessary to observe ontologies evolve until the population reaches a **stable state** from which their ontologies no longer evolve (regardless the object they encounter from the initially established environment, or agent they interact with from the initially established population) and communication is always successful (the success rate remains always ascending).

We are interested in answering the following main questions regarding the evolution of agent ontologies, from the initial state to the stable state: Does the defined experimental set-up lead to ontology evolution, regardless of the variable parameters and settings? How is evolution influenced by the parameters and variable settings defined above?

In order to respond to the previously defined research questions, we run experiments where we make these parameters and the initial conditions of the game vary, and observe the output. We have several experiments in which, for a certain parameter configuration, we run the game 10 times, generating new initial knowledge at the beginning of each game. The results are averaged over the 10 runs. The parameter configurations are the following: [3 agents, 1 feature x 2 values], [3 agents, 1 feature x 5 values], [3 agents, 2 features x 2 values], [3 agents, 2 features x 5 values], [3 agents, 4 features, 2x2+2x3 values], [4 agents, 2 features x 2 values], [5 agents, 2 feature x 2 values], [5 agents, 3 features x 2 values].

4.3 Experimental Results and Analysis

4.3.1 Convergence to the Stable State

In all the experiments we conducted, we observed that regardless of the controllable parameters and the randomness of the game (random initial ontologies, randomly selected objects and agents in each round), the population always converges to a **stable state** in which communication is always successful (from that point in time, success rate remains ascending) and agents no longer increase their number of categories. The stabilisation of the success rate sometimes occurs before reaching state **stable state**, in cases when ontologies continue to evolve trough intersection, but can no longer evolve trough adoption.

We show these characteristics in Figure 4.1, where we can see the success rates, respectively the average number of categories per agent, in an experiment with 10 differently initialized games, based on the same features. After less than 150 game rounds, the average number of categories stabilizes for each game. Also, after less than 150 rounds, the success rate remains ascending and asymptotically converges to 100%.

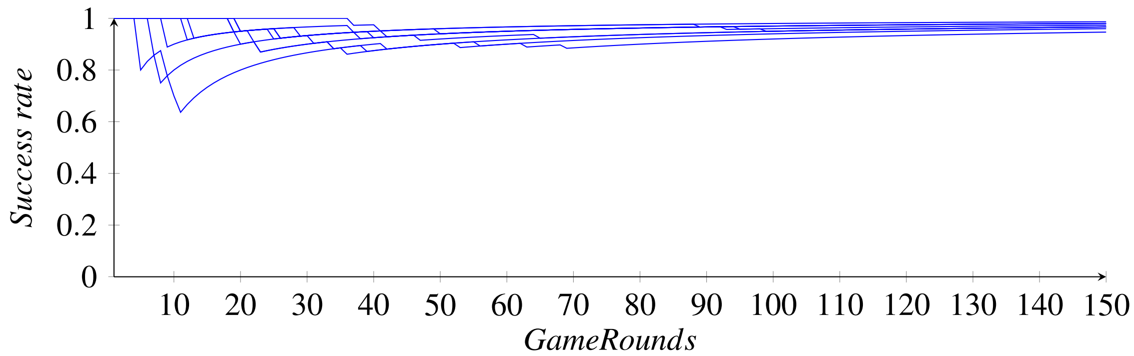
We see here that success rate evolution starts with the value 100%, as the first round of any game is always a success: an agent needs to intersect at least two times with another, to observe the other has a more precise classification.

The stabilization of the success rate coincides with the stabilization of number of categories adopted from other agents, but the categories created by intersection may evolve afterwards. We establish that the **stable state** is reached only after agents cannot evolve their ontologies, either by adoption or by intersection.

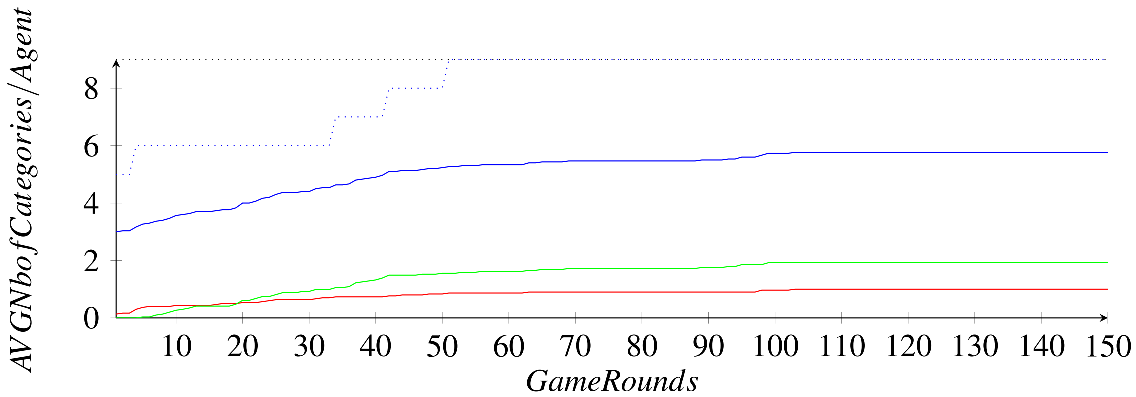
4.3.2 Does ontology evolution occur?

In this section we report on experiments that study the conditions of ontology evolution for the established protocol.

- Do agents always increase their number of categories? If not, which are these cases?
- What characterizes the output ontologies and the stable state?



Legend: x-axis: the number of game rounds played , y-axis: success rate of each of the 10 games, at current game round



Legend: blue = total number of categories,dotted blue=max number of categories, dotted black= maximum number of possible categories in the feature space red = number of categories created by intersection, green = number of categories adopted from other agents

Figure 4.1: 10 randomly initialized games, their success rates and their average number of categories/agent [3 agents, 2 features x 2 feature values]

- What are the environment changes that cause evolution to continue after a stable state?

Ontologies do not always evolve

As we saw in the experiments we conducted, in most cases, agents evolve their ontologies by playing the defined game. They do so by **creating intersection categories of existing ones** from their ontologies and **adopting categories from other agents**.

On average, evolving by creating intersection classes occurs faster than learning categories from other agents, especially in the first few rounds, as it only requires agents to encounter objects that belong to more than one most specific class in their ontology, whereas adopting categories depends on the rest of population and requires encountering objects related to previous classifications of the same agents, which are less frequent (Figure 4.1, Figure 4.3, Figure 4.5). The two operations boost each other, a new adopted category creates new intersection possibilities, and a new classification with a most specific category created by intersection can signal the interlocutor that the current agent has a more precise classification, thus leading to adoption.

Interestingly, there are cases in which the **population does not evolve**, all of them being conditioned by the fact that the agents cannot create any new categories by intersection:

1. when all agents have equivalent initial ontologies that cannot lead to intersection. For example, all the agents in the population have the initial ontology composed of *all objects*, *all white objects*, *all black objects*, *all white square objects*.

2. when all agents are equally able to distinguish between all the possible objects, and their categories cannot lead to intersection. For example: Pascal's ontology is composed of *Objet*, defined as *all objects* and *Blanc all white objects*.

   - Mary's ontology is composed of *Object*, defined as *all objects* and *Black*, defined as *all black objects*.
   - Mary's ontology is composed of *Object*, defined as *all objects*, *Black*, defined as *all black objects*, and *White*, defined as *all white objects*.

   In both cases defined above, for an environment containing only black and white objects, both Mary and Pascal are capable of differentiating between them. In both cases, Pascal will classify white objects as *Blanc*, and black objects as *Object*. Mary will always classify black objects as *Black* and white objects as *Object* in the first case, respectively *White* in the second case. In any case, the agents will always use the same corresponding labels.

Environment evolution enables ontology evolution

In the second example above, the population composed of the two agent ontologies will not evolve in the current environment that contains only *black* and *white* objects.

Let's introduce an object of color *red*. As they have not seen this colour before, agents do not use *red* value in their classification and do not have any category for it, so they both classify it as *Object*. Pascal would remember that, when he classified something as *Object* before, Mary has seen it as *Black*, which will lead to adopting a new category.

We have shown that if a new colour feature value is introduced in an environment, that the agents with stable ontologies did not have knowledge about, they are able to learn more categories from one another.

Different interactions result in different knowledge

We hypothesize that the **output** of the game depends on the **order in which agents interact** and the **objects** they observe. We have conducted an experiment with 4 agents and a feature space of 2 features with 5 values each. In this experiment, we generate the initial agent knowledge once, and run 10 games with the same initial knowledge. The resulted ontologies in the stable states differ from game to game.

As we see in Table 4.1, the final number of categories reached in the stable state varies from game to game with a significant variance value. We can conclude that the randomness of each interaction influences the evolution, agents obtaining different ontologies if they encounter agents and objects in a different order.

The following example explains the four possible outputs for the same initial ontologies: Mary's ontology contains concepts Object, Black and White,Pascal's ontology contains Objet,

	total number of categories in stable state										
agent	variance	number of categories in each game									
Agent1	3.12	32	32	34	35	31	36	36	33	33	35
Agent2	6.54	36	33	36	36	32	36	31	34	29	36
Agent3	4.9	39	32	36	36	29	33	33	32	33	33

Table 4.1: number of categories agents obtained in 10 games run with the same initial ontologies, [3 agents, 2 features x 5 values]

Blanc, and Hans's ontology contains Objekt. They interact in an environment that contains only black and white coloured objects.

- 1. in the first round, Hans and Pascal observe a white object, Hans sees it as *Objekt* and Pascal as *Blanc*.
  2. in the second round, they encounter a black object. Pascal tells Hans that it is *Objet*, and Hans, who sees it again as *Objekt*, acknowledges that Pascal makes a difference between the two objects and adopts Hans's categories. Hans's ontology becomes Objekt, Blanc_from_Pascal.

  Mary's ontology contains all the possible categories, and cannot evolve any more. Pascal's and Hans's ontologies are equivalent, and cannot evolve by intersection. Pascal and Hans cannot learn categories from Mary, because they are both able to distinguish between black (*Objet*, *Objekt*, *Black*) and white (*Blanc*, *Blanc_from_Pascal*, *White*) objects. Thus, the agents have reached the stable state.

- 1. a white object is classified by Hans as *Objekt* and by Mary as *White*
  2. a black object is classified by Hans as *Objekt* and by Mary as *Black*. Hans observes that Mary can distinguish between objects when he cannot, and adopts Mary's classes.

  Hans's ontology becomes *Objekt*, *White_from_Mary* and *Black_from_Mary*, equivalent to Mary's. Pascal's ontology cannot evolve anymore, as explained in the case above, thus agents have reached the stable state.

- 1. Hans classifies a white object as *Objekt*, and Pascal as *Blanc*
  2. Hans then classifies a white object as *Object*, and Mary as *Black*.
  3. Hans interacts with Pascal again about a black object, that he sees a *Objekt* and Pascal as *Objet*. Hans thus whishes to adopt Pascal's categories *Blanc* and *Objet*, and his ontologies becomes *Objekt*, *Blanc_from_Pascal*.
  4. Hans talks to Mary again about a black object, that he classifies as *Objekt* and Mary as *Black*. Hans remembers Mary's *White* from their previous interaction, for his same category *Objekt*, and adopts it.

  The output ontologies are equivalent to the case above, Hans's being *Objekt*, *Blanc_from_Pascal* and *Black_from_Mary*.

- In this last case, it is interesting to observe how **changes propagate** through the population:

1. a black object is seen by Pascal as *Objet* and Hans as *Objekt*
2. Hans then interacts with Mary, classifying a black object and then a white object, like in the second example. Hans's ontology becomes *Objekt*, *White_from_Mary* and *Black_from_Mary*
3. Pascal interacts with Hans again, about a black object. Because Hans has evolved, he now classifies it as *Black_from_Mary*. Pascal remembers his different classification from past, *Objekt*, and thinks that Hans can differentiate between the two black objects that he himself considers *Objet*. This way, Pascal evolves by trying to adopt Hans's categories *Objet* and *Black_from_Mary*. His output ontology becomes *Objet*, *Blanc*, *Black_from_Mary* and the agents have reached a stable state.

   One agent learns indirectly from another using an interlocutor: Pascal cannot learn new categories from Mary, as they are both able to distinguish between black and white object; Pascal learns the more specific category for black objects from Hans, who learned it from Mary.

Agents do not learn all the categories from each other

In the stable state, the agents did not necessarily have exchanged all their knowledge, their resulting ontologies being different in most of the cases. This is due to the fact that agents do not learn new classes, once all of them are able to discriminate between objects, even if some agents classifications may be more precise than others, as previously explained.

As we see in Tables 4.2 and 4.1 which display data from the same experiment, three ontologies that initially share 2 categories will result in ontologies of different sizes, but which share a very large percentage of categories.

number of categories the three agents have in common										
initial state	stable state									
2	32	27	34	35	23	33	28	28	26	32

Table 4.2: number of categories agents share over 10 games run with the same initial ontologies, [3 agents, 2 features x 5 values]

4.3.3 Influence of the parameters on ontology evolution

We suspect that by increasing the size of the problem (population size, feature space size) the game will generally require more rounds to converge to a stable state, as the probability of encountering the combination of agent pair and object that would lead to learning new categories decreases. The time required to converge depends also on the initial agent knowledge, and on the order of the random interactions. We conducted experiments varying these parameters (number of agents, number of features, number of feature values per feature), regardless initial agent knowledge.

	time to reach stable state												
population size	avg	min	max	time to reach stable state for each game									
3	58.5	32	103	32	69	40	65	53	44	50	93	39	103
4	94	15	161	95	161	155	15	117	113	48	52	96	88
5	226	71	475	406	288	475	103	190	71	95	288	196	149

Table 4.3: Time required to reach stable state (avegarge, minimum and maximum number of rounds) in differently initialized games with 3, 4 and 5 agents, on the same feature space (2 features x 2 values)

Population size

We conducted 3 experiments with 3, for and 5 agents where, for the same feature space (2 features x 2 feature values), we have played 10 games, each initialized randomly (so initial agent knowledge does not coincide from game to game).

The number of rounds required to converge varied largely from game to game, and the interval between minimum and maximum overlap between the three experiments, because it is highly influenced by the initial knowledge and randomness of each interaction. Even so, we can see in Table 4.3 that, on average, the number of rounds required for reaching the stable state increases with the population size.

This can be explained by the fact that, in order to learn categories by adoption, one agent needs to interact with a certain interlocutor and the probability of finding that interlocutor decreases with the increase of population size. One might expect adoption to occur at a much lower pace than intersection, with the increase of population size. This is not necessarily true, because the adoption of a new category creates possibilities for intersection, both the methods having a similar pace, as we see in Figure 4.2.

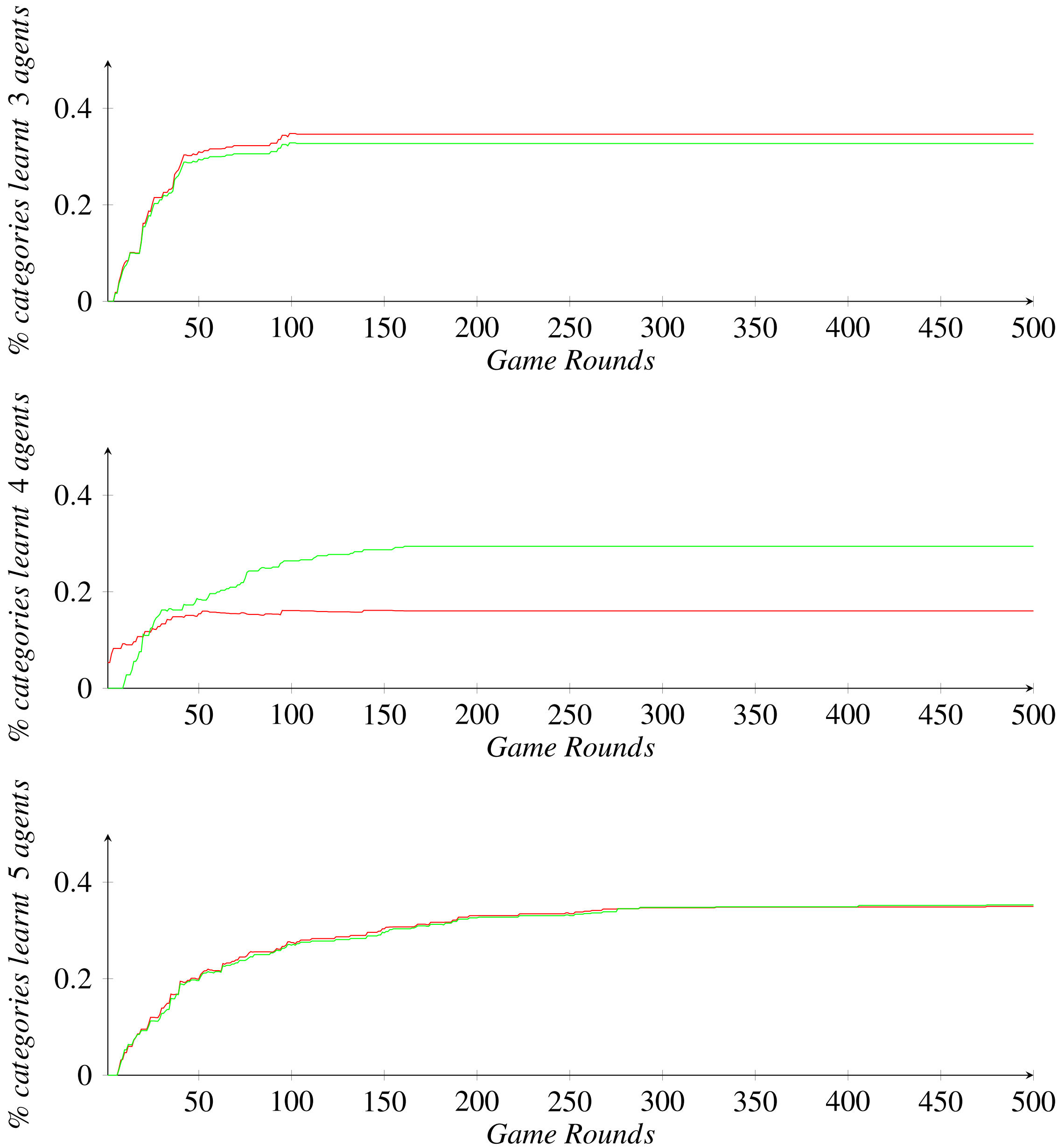
Feature Space size

The feature space determines the possible number of categories an ontology can have, the diversity of the initial ontologies and the diversity of the environment. Thus, we expect a larger number of categories to be learned in experiments with a higher feature size, but at a slower pace, as the probability of encountering an object that would lead to learning decreases.

We have conducted experiments in which for a population of 3 agent, we vary the feature space size, increasing both the number of features, and the number of values per feature.

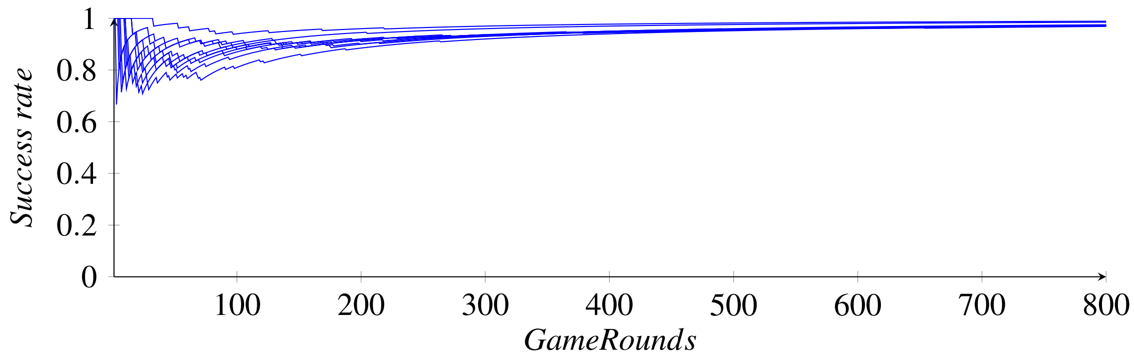
Increasing Feature Value size If we compare the results in Figure 4.1 where we defined 2 features with 2 feature values each, to Figure 4.3 where we increased the number of values per feature to 5, we can see that it takes 3 agents much longer to reach full success and to stabilize their ontologies. On average, the generated initial ontologies are larger in size due to the experiment settings. The average number of learned categories, both by intersection and adoption, are much larger, as detailed in Figure 4.4.

Increasing Feature size We have increased the feature space even more (4 feature, 2 of them with 2 and the others with 3 values), the results being shown in Figure 4.5. We also observe that the average and maximum number of categories an agent obtain in the 10 games are far lower than in games with a smaller features space. In 4.1 the total number of possible

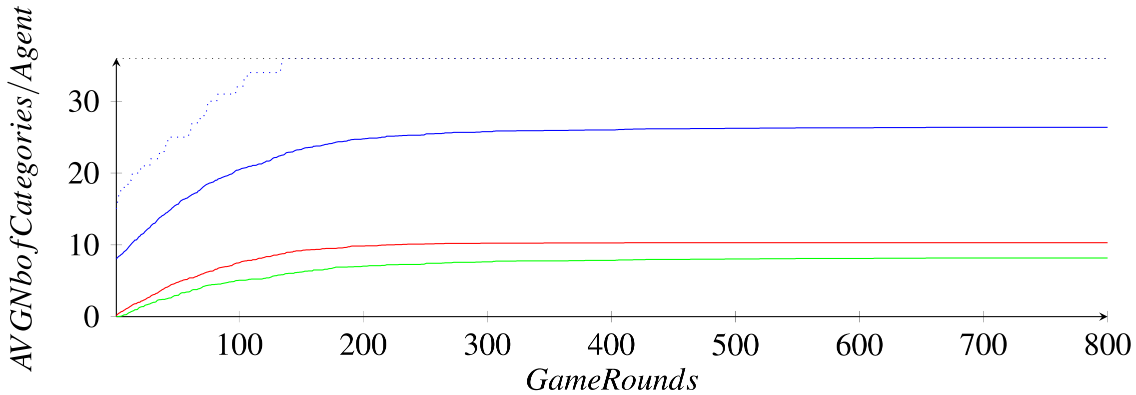


Legend: red= percentage of categories learned by intersection, green= percentage of categories learned by adoption

Figure 4.2: Percentage of the total number of categories of an agent, learned by intersection or adoption, averaged over 10 games played by 3, 4, respectively 5 agents



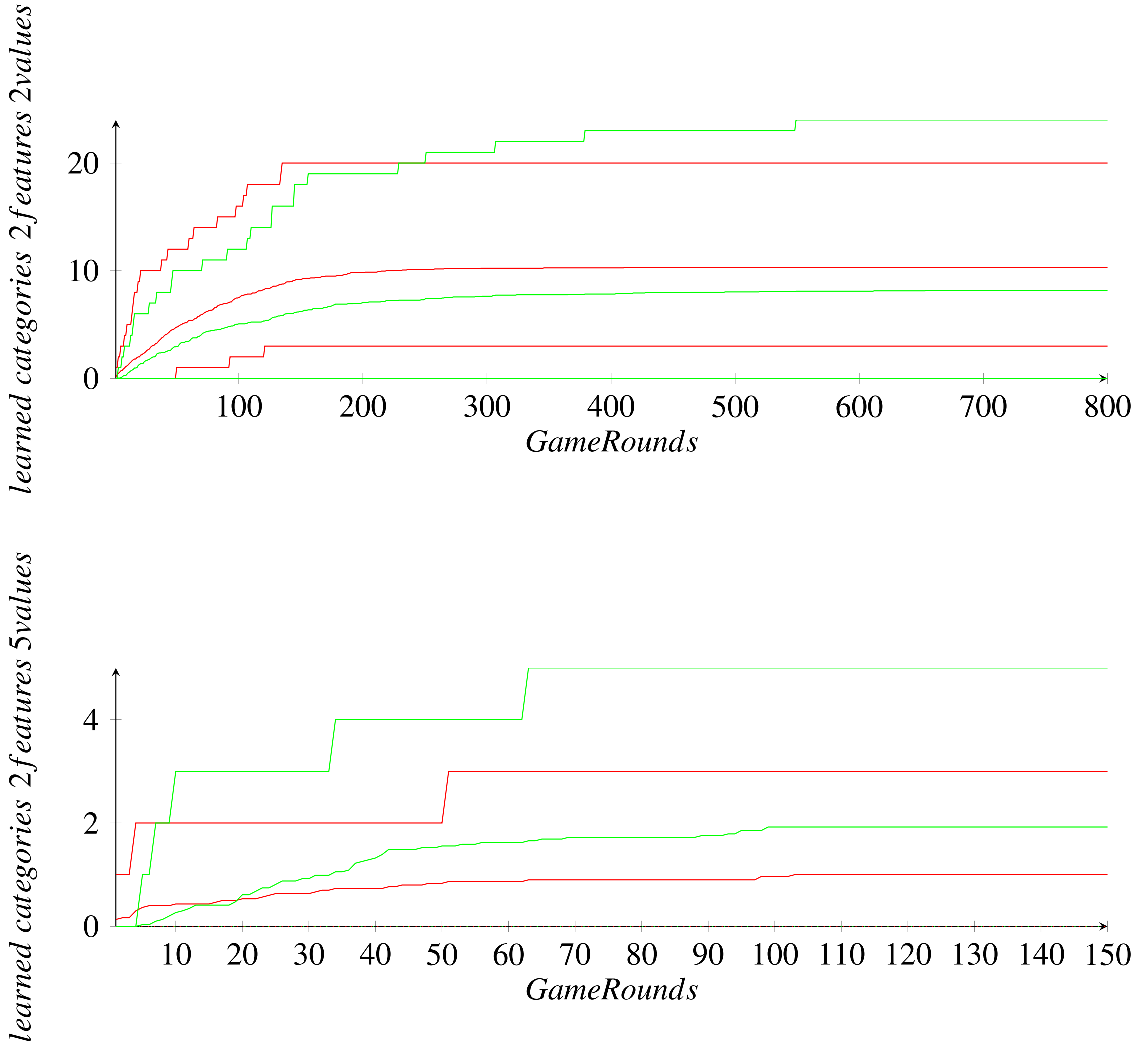
Legend: blue = success rate of each game



Legend: blue = total number of categories,dotted blue=max number of categories, dotted black= maximum number of possible categories in the feature space, red = number of categories created by intersection, green = number of categories adopted from other agents

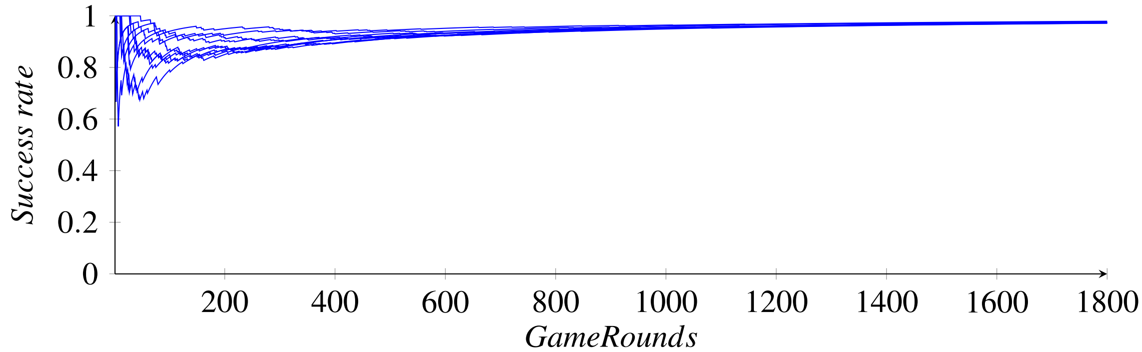
Figure 4.3: 10 randomly initialized games, their success rates and their average number of categories/agent [3 agents, 2 features x 5 feature values]

categories ($\prod i = 1n(|F_i| + 1)$) is reached 3 times, in Figure 4.3 it is reached 2 times, and here it is never reached. The higher the feature space complexity, the lower the probability that the initial ontologies and the random interactions will have a configuration favorable for obtaining all the possible classes for an agent, before reaching the stable state.

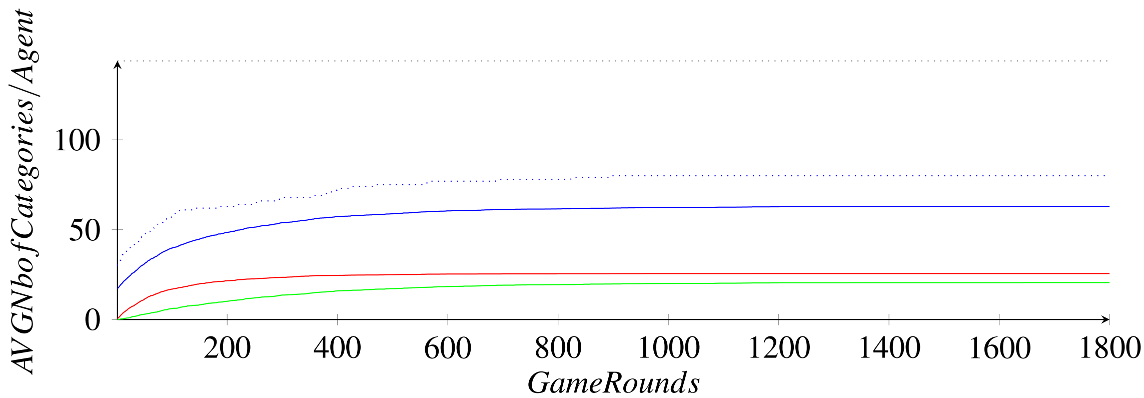


Legend: red = average, min, max number of categories created by intersection, green = average, min, max number of categories adopted from other agents

Figure 4.4: number of learned classes in 2 experiments - 10 randomly initialized games with 3 agents each - first experiment [2 features x 2 values], second experiment [2 features x 5 values]



Legend: blue = success rate of each game



Legend: blue = total number of categories, dotted blue=max number of categories, dotted black= maximum number of possible categories in the feature space, red = number of categories created by intersection, green = number of categories adopted from other agents

Figure 4.5: 10 randomly initialized games, their success rates and their average number of categories/agent [3 agents, 4 features (2x2 + 2x3 feature values]

— 5 —
Conclusions

5.1 Contributions and Achievements

We have reached our main goal, showing that agents can refine the classification precision of their ontologies while trying to improve communication. We have designed a simple cultural evolution protocol in which agents communicate about objects in the environment, invent new categories and adopt them from other agents in order to improve communication.

Conducting experiments with this protocol, we have shown that the population of agents is able to converge every time to a state of full communication success, where agents can no longer refine their classification precision. We have shown that indeed the agents have increased classification precision, acquiring new categories, and transmitting knowledge trough the population. The more or less precise adopted categories are combined to create more precise ones.

The resulting ontologies in this stable state are highly influenced by the randomness of game iterations, an initial ontology evolving differently after different games. Also, the population interacting does not necessarily reach the same ontology in the stable state, the resulting ontology depending on the initial knowledge and on the agent interactions.

5.2 Perspectives

Many more experiments can be performed using the proposed ontology evolution game, in order to investigate other aspects of knowledge. The game itself can be modified to allow a more complex representation of knowledge, or more complex interactions.

The current experiments have been run in a stable environment with equiprobable objects and agents. In the stable state, agents have already encountered all possible objects in the environment and agents in the population. Experiments need to be run to show how knowledge adapts to the **evolution of the environment**, by introducing new agents or objects with new feature values once the population has reached the stable state. We hypothesize that the agents will be able to learn new categories, until reaching a new stable state.

Knowledge about the environment is currently simplified, using only categories constructed as *conjunction* constraints on feature values. **disjunction** and **negation** can be introduced to obtain more complex ontologies.

The protocol of this game is based on the convention of having knowledge about the same features and using the same names for them. The game would become more complex if, instead of this convention, agents would have different knowledge about features, and use alignments between properties and between property values, that may or may not be correct. The alignments would have to be corrected dynamically trough communication, combining the game with [1].

The communication modality in the protocol can also be modified: the current *adoption* operation involves the communication of *complete* class definitions. We can experiment with reducing the **level of information** agents can communicate, and propose i) communicating class descriptions, deduced from the class definitions, that do not completely define the class ii) guessing the other agent's class definitions from the classification history. The two methods require the adopting agent to add a new class, guessing its definition, that may not correspond to the interlocutor's desired class.

Bibliography

[1] Jérôme Euzenat. First experiments in cultural alignment repair (extended version). In Valentina Presutti, Eva Blomqvist, Raphaël Troncy, Harald Sack, Ioannis Papadakis, and Anna Tordai, editors, *ESWC 2014 satellite events revised selected papers*, pages 115–130. Springer Verlag, Heidelberg (DE), 2014.

[2] Jérôme Euzenat and Pavel Shvaiko. *Ontology matching*. Springer-Verlag, Heidelberg (DE), 2nd edition, 2013.

[3] Christian Meilicke and Heiner Stuckenschmidt. Incoherence as a basis for measuring the quality of ontology mappings. In *Proceedings of the 3rd International Conference on Ontology Matching - Volume 431*, OM'08, pages 1–12, Aachen, Germany, Germany, 2008. CEUR-WS.org.

[4] Michael Spranger. *The Evolution of Grounded Spatial Language*. PhD thesis, Vrije Universiteit Brussels (VUB), Brussels, Belgium, 2011.

[5] Michael Spranger. Evolving grounded spatial language strategies. *KI - Kunstliche Intelligenz*, 27(2):97–106, 2013.

[6] L. Steels. *Experiments in Cultural Language Evolution*. Advances in interaction studies. John Benjamins Publishing Company, 2012.

[7] Luc Steels. Self-organization and selection in cultural language evolution. In Luc Steels, editor, *Experiments in Cultural Language Evolution*, pages 1 – 37. John Benjamins, Amsterdam, 2012.