# Size Matters

Grounding Quantifiers in Spatial Perception

**Simon Pauw**

# Size Matters

Grounding Quantifiers in Spatial Perception

ILLC Dissertation Series DS-2013-01

INSTITUTE FOR LOGIC, LANGUAGE AND COMPUTATION

For further information about ILLC-publications, please contact

Institute for Logic, Language and Computation
Universiteit van Amsterdam
Science Park 904
1098 XH Amsterdam
phone: +31-20-525 6051
fax: +31-20-525 5206
e-mail: illc@uva.nl
homepage: http://www.illc.uva.nl/



Cover design by Frans Mettes.
Printed and bound by Off Page. ISBN: 978–94–6182–336–6

# Size Matters

Grounding Quantifiers in Spatial Perception

Academisch Proefschrift

ter verkrijging van de graad van doctor aan de
Universiteit van Amsterdam
op gezag van de Rector Magnificus
prof.dr. D.C. van den Boom
ten overstaan van een door het college voor
promoties ingestelde commissie, in het openbaar
te verdedigen in de Agnietenkapel
op vrijdag 1 november 2013, te 14.00 uur

door

Simon Pauw

geboren te Amsterdam, Nederland

| | |
|---|---|
| Promotoren: | Prof. dr. F.J.M.M. Veltman |
| | Prof. dr. L.L.L. Steels |
| Overige leden: | Prof. dr. ir. R.J.H. Scha |
| | Prof. dr. M.J.B. Stokhof |
| | Dr. R.A.M. van Rooij |
| | Dr. P.A. Vogt |
| | Dr. J.H. Hilferty |

Faculteit der Natuurwetenschappen, Wiskunde en Informatica

# Contents

# Acknowledgments

I would like to thank Michael Spranger, Martin Loetzsch, Joe Hilferty and Nancy Chang with whom I have collaborated intensively throughout the past few years. Without them I would not have gotten far and it certainly would not have been as much fun. I would also like to thank both my supervisors Frank Veltman and Luc Steels for their support and feedback, and providing the framework and financial means to conduct this research. Furthermore I would like to thank Remko Scha, Remi van Trijp, Lucas Beerekamp, Ranne Hovius, Daniel Oberski, and Frans Mettes (who also designed the cover of this thesis) for providing feedback and corrections; my paranymphs Rogier Heumakers and Emma Pauw; and Joris, Pieter, Joachim, Katrien, Vanessa, Katya, and Oscar, my collaborators from the Vrije Universiteit Brussel, Sony CSL Paris, and the Universitat Autònoma de Barcelona.

Amsterdam Simon Pauw
August, 2013.

# Part I

# Introduction

# Chapter 1

# Preface

## 1.1 Introduction

Human language is the product of *cultural evolution*. This is a point of view about language that is not accepted by all of its students, but it is a point of view that I share with other researchers in *evolutionary linguistics*.

Language is not a static collection of words and grammar rules, but a complex adaptive system. It changes over time in a way not unlike biological evolution. Speakers of a language continuously create variations: they make mistakes, they try to convey novel concepts. Some of these variations are more successful in communication than others. The most successful variations will continue to be used, other variations disappear. Just as the principles of biological evolution explain how species continuously adapt to their environment, these principles of cultural evolution do the same for language.

This might sound like a truism. In a post-Darwinian era, can we conceive of any other view? But, there are quite a number of prominent linguists and philosophers who can. Most notably, Noam Chomsky and Jerry Fodor, who take a nativist stance: we (humans) have an innate universal grammar, a dedicated mechanism in the brain that contains the basic structures of all languages[1] (Chomsky, 1986; Fodor, 1983). So, there is a task for empiricists (that favor an evolutionary view) to show that universal grammar is not a necessary assumption, that all aspects of language can be explained as the result of cultural evolution.

Lacking empirical data, the most common methodology in this field is the use of computer simulations. How can we show that language is the result of cultural evolution? By repeating it. And this time we pay attention. Of course it

[1]The main argument of such a view is that without a universal grammar, languages would not be learnable. If language is not constrained by a universal grammar, the discrepancy between the amount of possible languages and the data we have for learning them is too vast. We would never be able to learn a language. This is often referred to as the *Poverty of the stimulus* argument.

is not feasible, using computational simulations, to replicate the entire evolution of language, from its genesis to the current state of every language. So, typically researchers in the field focus their attention on one particular aspect of language. The earliest experiments mainly argued against a nativist view by showing that in principle it is possible for a language to emerge as a result of cultural evolution (Steels, 1999, 2008) and that languages will automatically evolve to be learnable (Kirby and Hurford, 2002).

But, the languages that evolved in those experiments do not get close to the complexity of real human languages. In search of more realism, recent experiments limited their focus to specific language domains. These experiments do not simulate the evolution of language from scratch, but assume a primitive existing language and isolate the emergence of one specific aspect of language (such as language to express color (Bleys et al., 2009), spatial relations (Spranger, 2011), or grammatical agreement (Beuls and Steels, 2013)).

Presently, I think we can see another step towards realism being taken, a new development of which this dissertation is a product. Some recent studies have started to look more carefully at the evolutionary paths themselves (van Trijp, 2012; Beuls and Steels, 2013). Words and grammatical constructions do not simply pop up in a language and then stay put. They come to exist as derivations of other words and then start to live their own life: they gradually change in function, meaning, and form. For example, the English conjunction *while*, stems from the Old English noun *kwīl* "time". This example fits a very common pattern in languages called *grammaticalization*. Grammaticalization describes the tendency of words to become more grammatical. The Old English *kwīl* had a clear semantical content, whereas *while* has mostly a grammatical function in English.

The process of grammaticalization is one of the central themes of this dissertation. I use the example of *gradable quantifiers* (words such as *many* and *few*). In English these gradable quantifiers belong to the grammatical class of quantifiers (just as *one*, *all*, or *half*). But they did not start out like that. For example, historical linguists established that the word *few* finds its origins in the Old-English *feawe*, which was not a quantifier, but an adjective that could also mean "small". In the last two chapters of this thesis I describe a cognitive mechanism that accurately simulates this grammaticalization path called *reanalysis*.

This reanalysis mechanism is a general mechanism that can explain the grammaticalization of any word into any grammatical category. But, not all words follow the same grammaticalization path. Some adjectives emerge as adjectives and stay that way (such as the word *big*), some quantifiers have different origins (for example *several* derives from the Latin verb *seperare* "to separate"). When explaining why *many* and *few* follow this particular evolutionary path, we also need to explain why other words don't. In this dissertation I hypothesize that this might be due to the unique cognitive properties of these gradable quantifiers.

Psycholinguists have shown that there is a close cognitive relationship between

size and number. Judgments of size (underlying modifiers such as *big* and *small*) depend on perceptual features of objects (or sets of objects) in the environment. Judgments of approximate number (underlying terms like *few* and *many*) exploit a combination of spatial features that apply exclusively to sets of objects, such as their size and density (Durgin, 1995). This cognitive overlap between the concepts of size and number can account for this particular grammaticalization path: the dependence on size motivates their adjectival origins, while their application to sets of objects works as a syntactic magnet that draws them into the grammatical category of quantifiers.

Depending on your generosity, there are two ways to interpret the results of this dissertation: The stingy interpretation is that I show very accurately how and under which cognitive assumptions gradable quantifiers might emerge in a language. A more generous interpretation is to view the gradable quantifiers as a test case, leading to a better understanding of the process of grammaticalization. Furthermore, this work illustrates how the inclusion of data from psycholinguistics may illuminate the grammaticalization patterns found by historical linguists. Of course, I do not claim that this thesis has the final word on grammaticalization, or on quantifiers, or on language evolution for all that matters. The goal of evolutionary linguistics is to show that human languages are the product of cultural evolution. What I hope to convince you of is that using computational models of language evolution to bridge the gap between psycholinguistics and historical linguistics brings us closer to that goal.

## 1.2 Outline

Except for the present chapter and Chapter 2, all chapters in this thesis either have been published already or will be published in the near future as independent articles. This means that the chapters can be read (or ignored) on their own, and they can be read in any order. The organization of the chapters in this dissertation is therefore mainly thematic. The dissertation is divided into three parts. The first part of this dissertation contains this introduction and a discussion of the literature on cognition of gradable quantifiers. The second part of the dissertation consists of three papers that discuss my contributions to the technical machinery that is needed for conducting robotic language experiments. The final part consist of three papers that describe the experiments themselves.

### 1.2.1 Part 1

There are many factors influencing the interpretation of gradable quantifiers: the size of the type of object being quantified over, their density patterns, some expected norm, etc. There is a vast amount of literature from different scientific disciplines, addressing a variety of cognitive aspects of these gradable quantifiers.

**Chapter 2** provides an overview of the results of these different disciplines. It is abundantly shown that spatial features such as size and density are essential for the judgment of gradable quantifiers.

### 1.2.2 Part 2

Developing robotic experiments requires an enormous amount of technical work that is normally skimmed over in papers on evolutionary language experiments. This part of the dissertation is dedicated to my contributions to the technical machinery.

**Chapter 3** describes a fully operational procedural semantics that is created for robotic communicative interactions called Incremental Recruitment Language (IRL). IRL contains a number of mechanisms that are needed for conceptualization. The goal of this chapter is to provide a detailed overview of the most essential IRL mechanisms.

Grounding language in sensorimotor spaces is an important and difficult task. In order for robots to be able to interpret and produce utterances about the real world, they have to link symbolic information to continuous perceptual spaces. This requires dealing with inherent vagueness, noise and differences in perspective in the perception of the real world. **Chapter 4** presents two case studies for spatial language that show how cognitive operations—the building blocks of grounded procedural semantics—can be efficiently grounded in sensorimotor spaces.

**Chapter 5** studies how quantificational expressions such as *few*, *three* and *all* can be grounded in real-world perception. I discuss a computational model, called *clustering quantification*, designed for use in robot-robot interaction scenarios which involve discrimination tasks for objects in the real world. The performance of this model is compared with an alternative type-theory based model. It is shown that clustering quantification is more suitable for real-world applications.

### 1.2.3 Part 3

This last part of the dissertation discusses the language evolution experiments themselves.

The gradable quantifier *many* does not refer to the same amount in the utterances "many students in the classroom" and "many teachers in the classroom". The interpretations of such quantifiers depend on an expected frequency (a norm)—normally we expect there to be more students than teachers in a classroom. **Chapter 6** provides a cultural-evolution explanation for the emergence of norm-dependent quantifiers, focusing in particular on the role of environmental constraints on strategy choices. Through a series of situated interaction experiments, we show how a community of robotic agents can self-organize a quantification system. Environments in which the distribution of objects exhibits

some degree of predictability creates favorable conditions for context-dependent quantifiers.

**Chapter 7** discusses a series of experiments in which it is shown that the adjectival origins of gradable quantifiers can be explained by the cognitive overlap between these quantifiers and adjectives such as *big* and *small*.

**Chapter 8** illustrates the grammaticalization of gradable quantifiers into quantifiers. It is shown that other existing quantifiers can create attractor positions for other modifiers to grammaticalize into and that *many* and *few* follow this path in search of reducing cognitive effort. Thus, arguing that the shift from qualifying to quantifying expression has a cognitive motivation.

# Chapter 2

# The Idea of Many - A Literature Overview

Gradable quantifiers—i.e., words like *many*, *few*, *much* and *little*—express quantity (or *numerosity*). And, the way they do this is radically context dependent. Most people would consider a hundred pages *many* for an article but not for a novel. There are many factors influencing the interpretation of gradable quantifiers: the size of the type of object being quantified over, their density patterns, some expected norm, etc. The way these factors influence the interpretation of gradable quantifiers might teach us something about their cognitive status. There is a vast amount of literature from different scientific disciplines, addressing a variety of cognitive aspects of these gradable quantifiers. As of yet, little effort has been made to compare the results of these different disciplines with each other in one overview. The aim of this chapter is to provide such an overview.

## 2.1 Introduction

Gradable quantifiers like any other quantifier, are used to express *numerosity* (the number of elements in a set). This is not necessarily the only role that quantifiers have—they are also used to mark definiteness, aspect, deixis, and inferential patterns—but as the name already suggest, numerosity is an essential semantic dimension of quantifiers.

Many attempts to model the semantics of quantifiers, define the quantifiers over the cardinality of sets (Barwise and Cooper, 1981). In some cases, this mapping can take the context dependence (Fernando and Kamp, 1996) and/or the vagueness of these quantifiers (Zadeh, 1983) into account. But, regardless of their specific flavor, all these models presuppose the accessibility of the cardinality of the sets that are being quantified over. However, studies in psycholinguistics (Hormann, 1983; Moxey and Sanford, 1993b), psychophysics (Bevan et al., 1963; Krueger, 1972; Durgin, 1995) and developmental psychology (Piaget, 1952; Mix

et al., 2002) show that this assumption does not correspond to human cognition. For big sets, humans do not perceive the number of a set of objects directly as a feature of that set. Rather, we perceive a number of spatial features (such as extent, density, contour, ...) that are used together to estimate the numerosity.

Roughly speaking there are three fields of research that study these issues in relative isolation: The field of *psychophysics* (discussed in the next section) has produced a number of studies on numerosity perception (regardless of its effect on language). Studies in *developmental psychology* (Section 2.3) show how infants acquire numerosity perception. And studies in *psycholinguistics* (Section 2.4) show how contextual parameters influence the interpretation of gradable quantifiers. To the best of my knowledge there is no overview of all these studies. This chapter aims to provide such an overview and show the remarkable overlap in results from these different disciplines.

## 2.2 Numerosity perception

An utterance like "many blocks" expresses a vague amount. In order to understand the cognition of gradable quantifiers, we first need to understand how humans perceive amount. This section provides an overview of the literature on this *numerosity perception*.

Jevons (1871) argues that numerosity perception can not be attributed to one single cognitive mechanism, but rather involves a quite heterogeneous set of cognitive mechanisms. It is clear that there is a distinction between explicit counting and estimating the number of objects. But, Jevons found another distinction: For small sets (up to four elements) humans can instantly and accurately determine the number of items. Beyond this range, accuracy drops significantly, suggesting a different cognitive mechanism for estimating the number of small sets (now referred to as *subitizing* (Kaufman et al., 1949)). Later experiments have systematically confirmed these findings (see Mandler and Shebo (1982) for an overview).

It is now generally agreed upon to distinguish three separate cognitive mechanisms that are involved with numerosity judgment: Subitizing, a very accurate fast process that only works for small amounts (four or less); the approximate number system (ANS) Halberda and Feigenson (2008), a fast but inaccurate process of estimation; and a very slow but accurate process of explicit counting. Most of the experiments that establish the different mechanisms involve reaction time measurements. The different mechanisms are associated with different reaction time curves.

For example, Figure 2.1 shows the graphs from an experiment presented in Mandler and Shebo (1982). In this experiment, participants where presented with arrays containing between one and twenty dots. They were asked to report the number of dots in the array. The reaction time graph shows 3 different parts:

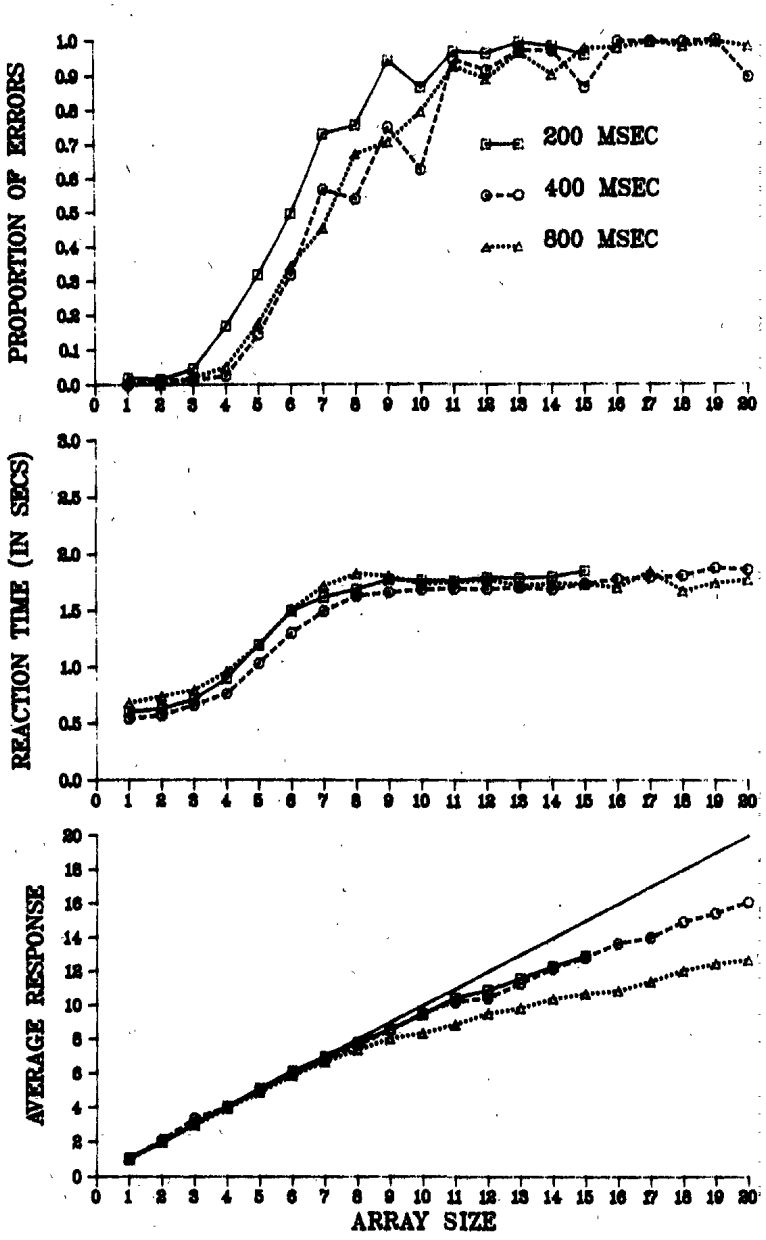



Figure 2.1: Results reaction time experiment from Mandler and Shebo (1982) for numerosity judgments under three exposure conditions (200, 400, and 800 msec) for proportion of incorrect responses (top panel), reaction times (middle panel), and average number response (bottom panel).

Up to three of four items the reaction time does not increase much, which is associated with subitizing. After that there is a linear increase of reaction time (explicit counting). After seven or eight items, the reaction time stops increasing which is associated with number estimation (ANS). It is also shown that as soon as people start estimating the number of items, the error increases dramatically due to systematical underestimation (confirming the findings of Kaufman et al. (1949)). Following the same methodology, Minturn and Reese (1951) show that subitizing is indeed a fundamentally different process from counting, and Taves (1941); Trick and Pylyshyn (1991) confirm the difference between subitizing and number estimation[1].

For the remainder of the section I will be focussing on the research concerning the approximate number system, the part of numerosity perception that is most directly related to gradable quantifiers.

### 2.2.1 Perceptual features of approximate number

Most research in the field of ANS shows that approximate numerosity estimation is not unitary. That is to say, the number of elements in a set is not perceived as a single feature, but there are a number of perceptual features (e.g., density, size, clustering and contrast) that are combined in estimating the number of objects. In order to identify the perceptual features that are involved with numerosity judgment most of these studies compare identical quantities in different spatial arrangements.

A clear results of this approach is the strong influence of extent (or more specifically, *area size*) on the perceived numerosity of a set. For example, Krueger (1972) shows that the perceived numerosity of a group of dots is influenced by size of the surface area that the group of dots occupies. In this study, participants are confronted with a number of dots (between 25 and 200) scattered over area's of different sizes ($56cm^2$, $225cm^2$ and $900cm^2$) and asked to estimate the number of dots. The participants systematically underestimate the amount of dots in all trials. But the reported estimations were much closer to the actual number of dots for the larger surface area's. These results confirm similar findings of Bevan et al. (1963). There, the effect of numerosity judgement was tested using a jar of beans rather than dotted surfaces.

The other important spatial influence that has been found is that of *density patterns* (as initially hypothesized by Barlow (1978))—i.e., how seemingly closely objects are clustered together in a set. Barlow (1978) do not experimentally test this hypothesis. To the best of my knowledge, only Durgin (1995) and Durgin (2008) provide experimental data. He confirms the hypothesis by showing that lower perceived density results in lower numerosity judgements. The relatively

[1]Trick and Pylyshyn (1993) argue that subitizing is related to mechanism in the brain called FINST, a preattentive mechanism that parallel tracks objects in our visual environment.

small number of targeted experiments might be due to the difficulty of isolating the density feature (i.e., controlling for influences of other spatial features such as individual object size and total area size). Durgin does this by relying on so called after-effects: if a trial display filled with dots is preceded by a display with a much higher density, the density of the trial display seems much lower in density. This way the perceived density can be manipulated without affecting the perception of any of the other spatial features. Using these after-effects, Durgin goes on to show that the numerosity judgments are much lower for displays with a seemingly low density.

Other influences show that the picture might be more complicated. For example, Ginsburg (1978) shows that item arrangement influences numerosity judgments: more randomness leads to higher judgments. And, Bevan and Turner (1964) show that figure/ground perception has a significant influence on numerosity judgment. I.e., if the objects and their container are seen as a whole (i.e., the container is part of the figure), there is a strong positive correlation between area and perceived number (confirming Krueger (1972)). However if the container is explicitly marked as ground, a reverse effect is found.

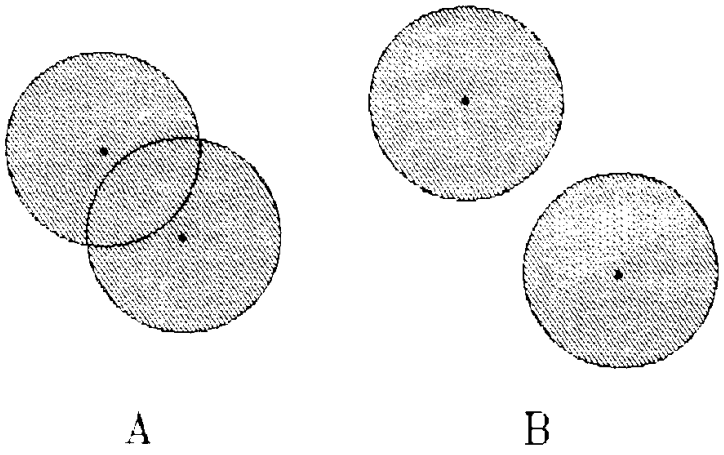



Figure 2.2: Occupancy model from Allik and Tuulmets (1991). Dots are hypothesized to occupy a region. Perceived numerosity is determined by the total area occupied by the dots. When regions of dots partly overlap occupancy regions overlap, the perceived numersoity is lower.

These last results might create a slightly more complicated picture than the area/density experiments would have you believe, but they do not deny the area/density effects. Durgin (1995) even suggests that increased randomness might increase the perceived density and therefore the effects found by Ginsburg (1978) can completely be interpreted within the area/density picture.) There is

however one study (that I am aware of) that explicitly opposes the multidimensional view of numerosity: Allik and Tuulmets (1991) argue for a unidimensional model of numerosity which they call the occupancy model. In this occupancy model, every element (dot) occupies a region. This region extends with a radius $R$ around the actual object. The numerosity is determined by the total area occupied by the dots. Overlapping regions causes a smaller total area, and therefor a smaller judged number. See also Figure 2.2. An unlikely model, according to Durgin (1995). He argues that the model might fit the data provided in Allik and Tuulmets (1991), but in order to fit other data (i.e., the data from Durgin (1995)) the radius $R$ has to vary with the density patterns, thus making density an implicit factor in this model.

## 2.3 Results from Developmental Psychology

Another source of evidence that supports the multidimensional view on numerosity judgement comes from developmental psychological research. For example, Pufall et al. (1973) argue that numerosity is a multidimensional concept and that children have to learn how the different dimensions (extent and density) work together. The research in this field essentially makes two points: 1) infants do not have an innate number concept, but they do have an innate density and extent concept; and 2) this is because the number concept is a high-level concept that relies on learning to combining these lower level concepts of extent and density.

### 2.3.1 Innate Numerosity Perception

The first point has extensively been discussed by Mix et al. (2002); Clearfield and Mix (2001). They argue that infants can not discriminate arrays of dots based on the number of items in them, they only respond to differences in density, length (extent) and contour length. The experiments they report involve infants of 4 to 12 months, much too young to directly ask if they see any differences between arrays. This problem is circumvented by using so called habituation experiments. Habituation experiments rely on the fact that infants, presented with a number of arrays, tend to look longer at an array that does not fit a given pattern (i.e., a surprising result). So infants are presented with a series of habituation arrays that keep one variable constant (e.g., density) followed by a test array that either fits (has the same density) or breaks (has a different density) the pattern. If the test array that does not fit the pattern results in longer looking times, than apparently the infants are sensitive to differences of the variable in question. Figure 2.3 shows an example.

Mix et al. (2002) show that infants indeed do respond to changes in density, extent and contour, but if properly controlled for these spatial constraints, no habituation effects can be found. There is a number of earlier experiments that has

| Control | Density | Length |
|---|---|---|
| Habituation Array 1 | • • | • • • |
| Habituation Array 2 | • • | • • • |
| Test Array | • • • | • • |

Figure 2.3: Habituation trials from Clearfield and Mix (2001) used to test numerosity perception controlling respectively for density and extent.

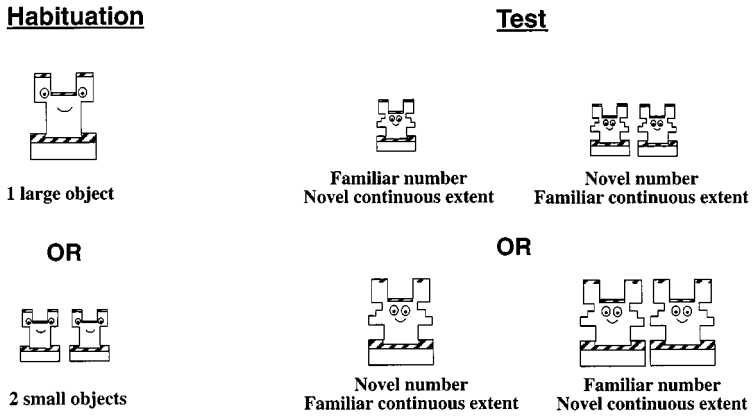



Figure 2.4: Habituation trials from Feigenson et al. (2002) controlling for extent and density.

shown the contrary (e.g., Starkey and Cooper (1980); Antell and Keating (1983); Cooper (1984); Starkey et al. (1990) all find habituation effects for number), but as Mix et al. (2002) argue, none of these experiments manage to properly control for the spatial features. Feigenson et al. (2002) repeated the experiments presented in Starkey et al. (1990) but controlled for extent and density by making the size of the objects variable (as shown in Figure 2.4). They show that by properly controlling for these spatial features, the number habituation effects disappear.

### 2.3.2 Development of Numerosity Perception

So, if the findings of Mix, Clearfield and others are correct, infants do not have an innate numerostiy concept. These findings are in line with data from developmental psychology (Piaget, 1952, 1968; Pufall et al., 1973). In these studies it is argued that children acquire the number concept by incrementally learning to coordinate extent and density (or "crowding", as Piaget calls it). Piaget (1952) identifies three different stages in which the numerosity concept is learned:

1. The child judges numerical relations purely in terms of "global" similarities (i.e., extent or density) (Piaget, 1952). In other words, a row of chips is judged as "more" if it is longer. Later studies show that this stage actually comprises two substages (Mehler and Bever, 1967; Piaget, 1968):

   (a) The child uses the most salient feature (density or extent) for comparing sets. The used feature can differ between trials. The reported ages for this stage differ: 2y3m-3y0m (Piaget, 1968); 2y6m-3y2m (Mehler and Bever, 1967); 3y (Pufall and Shaw, 1972; Pufall et al., 1973).

   (b) The child concentrates on one single feature across trials (i.e., extent). The reported ages for this stage differ: 3y0m-3y10m (Piaget, 1968); 3y2m-4y0m (Mehler and Bever, 1967); 4y (Pufall and Shaw, 1972; Pufall et al., 1973).

2. The child coordinates both length and density. However, this is only done visually, the child cannot reason about it. At this stage, if the child has to compare arrays where length and density are inversely related, it will revert to using only one single feature. (Piaget, 1952, 1968; Pufall and Shaw, 1972; Pufall et al., 1973).

3. Numerosity perception is completely developed (Piaget, 1952, 1968; Pufall and Shaw, 1972; Pufall et al., 1973).

Similar to the experiments described above, the number conception of the children is tested using array's of dots or chips. The children are asked to report which array contains more dots. Different configurations are used to expose the

stage of numerosity perception of the child. For example, figure 2.5 shows the different types of configurations used by Pufall and Shaw (1972). Which configurations are correctly assessed depends on the development stage of the child. For example, a child in stage 2 will have no trouble getting Configurations 1-3 right, but Configuration 4 is problematic since the inverse relation between length and density will make it focus on only one of the two features. Some minor difference in ages and granularity of stages aside, Piaget (1952, 1968); Pufall and Shaw (1972); Pufall et al. (1973) all show that in earlier stages, children tend to focus on only one dimension (i.e., length or density) and only later learn to coordinate them into a fully developed numerosity concept (at around the age of 7).

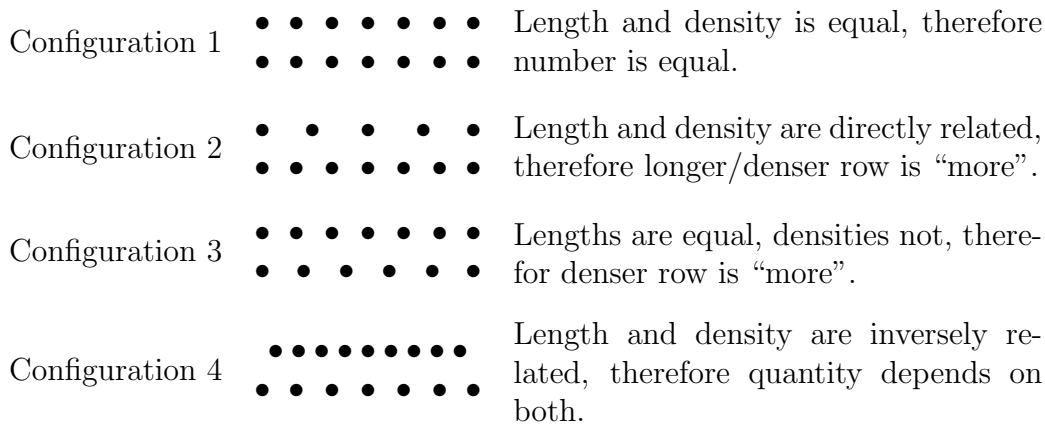



Figure 2.5: Configurations used by Pufall and Shaw (1972) to test the child's numerosity perception.

Piaget suggests that the different stages are the result of learning which transformations preserve the number of a set (e.g., moving objects more closely together) and which transformations change the number of objects (e.g., adding or removing objects)—i.e, *conservation principles*. If this is true, the question remains how these conservation principles are learned. Strauss and Curtis (1981); Starkey et al. (1983) independently found that within the subitizing range infants (of respectively 10-12 months and 7 months) do in fact already perceive numerosity. It could be that perception of these low numerosities provides the seed for further development of numerical skills by establishing Piaget's conservation laws (Marmasse et al., 2000). Additionally, Wohlwill and Lowe (1962) speculate that children learn conservation of number via counting[2]. Children learn to count well before they learn to conserve. It might be that by counting the elements in sets that are being manipulated, children learn to understand the relation between density, extent and number.

[2]The experiments described in Wohlwill and Lowe (1962) neither confirm nor refute this so called *reinforcement hypothesis*.

## 2.4 Results from Psycholinguistics

The previous sections showed the effects of spatial features on numerosity perception of the approximate number system (ANS). Most studies suggest that numerosity perception requires the coordination of extent and density features. Since gradable quantifiers rely on ANS, we expect some of the particularities of ANS to affect the interpretation of gradable quantifiers. This section discusses a series of studies on the judgment of gradable quantifiers. The studies expose a number of factors that influence the judgment of gradable quantifiers, such as expected frequency, size of the individual objects, area size of the entire group, figure/ground effects and contrast. Though none of these studies explicitly link these effects to ANS, some of them would be very hard to interpret in any other terms (such as the effect of surface area on the judgment of gradable quantifiers).

### 2.4.1 Expected Frequency (Norm)

It is not difficult to find anecdotal evidence that the interpretation of gradable quantifiers depends on some prior expectation. Clearly the utterances "many people read E. L. James' latest novel" and "many people read this thesis" do not refer to similar amounts of people. Their interpretations depend on a prior expectation we have about the number of readers in both cases. It is however quite hard to come by experimental data confirming this intuition. To the best of my knowledge, only Moxey and Sanford (1993b) explicitly addresses this issue. In one of the experiments that the chapter describes, participants are presented with three different snippets:

> The residents' associations Christmas party was held last night in the town hall. QUANT of those who attended the party enjoyed what might be called the social event of the year. [RA condition]
>
> At yesterdays party conference, Mr Cameron spoke about the effects of education cuts on British universities. QUANT of his audience were convinced by his conclusions. [PC condition]
>
> A survey has recently been carried out to find out whether or not female students prefer to be examined by female doctors. QUANT of the local doctors are female. [S condition]

Where QUANT can be any of the quantifiers *very few*, *few*, *only a few*, *not many*, *a few*, *quite a few*, *quite a lot*, *many*, *a lot*, and *very many*. After hearing the snippet, the participants were asked to report the estimated proportion of people who went to the party/were convinced by Cameron/preferred female doctors. The reported proportions where compared to independently established prior expected proportions. The results are shown in Figure 2.6.

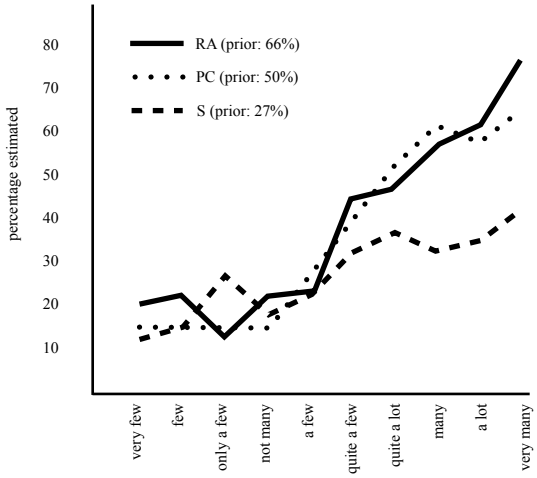



Figure 2.6: Results from Moxey and Sanford (1993b). The influence of context on the interpretation of quantifiers. The lines are for presentational convenience; the groups from which each point is derived are independent. The quantifiers are ordered *post hoc* in terms of ascending mean values.

For high-magnitude quantifiers (*quite a few*, *quite a lot*, *many*, *a lot*, and *very many*) the results clearly illustrate the effect of different prior expectations on the interpretation of these quantifiers. For low-magnitude quantifiers (*very few*, *few*, *only a few*, *not many*, and *a few*) the results are not significant. Moxey and Sanford attribute this to the floor effect: the proportions that are described by the low-magnitude quantifiers are so small that variations are not measurable in this setting.

That the prior expectation about the context is an important factor in the interpretation of gradable quantifiers is undeniable. Hormann (1983) goes even so far to suggest that without context gradable quantifiers are entirely meaningless. He shows this by asking participants to report the number of paperclips they expect after hearing the contextless phrase "QUANT Büroklammern" ("QUANT paperclips"), where QUANT is one of the quantifiers *ein paar* (*a few*), *einige* (*some*) or *mehrere* (*several*). The quantities reported by the participants make it impossible to distinguish the quantifiers: the resulting histograms are almost identical for all the quantifiers. However, by providing the verbal context "auf them Tisch" ("on the table"), the interpretations of the different quantifiers diverge. See, Figure 2.7 for the results. So, the context information of these quantifiers is so vital for their interpretation that without it, they become entirely meaningless.

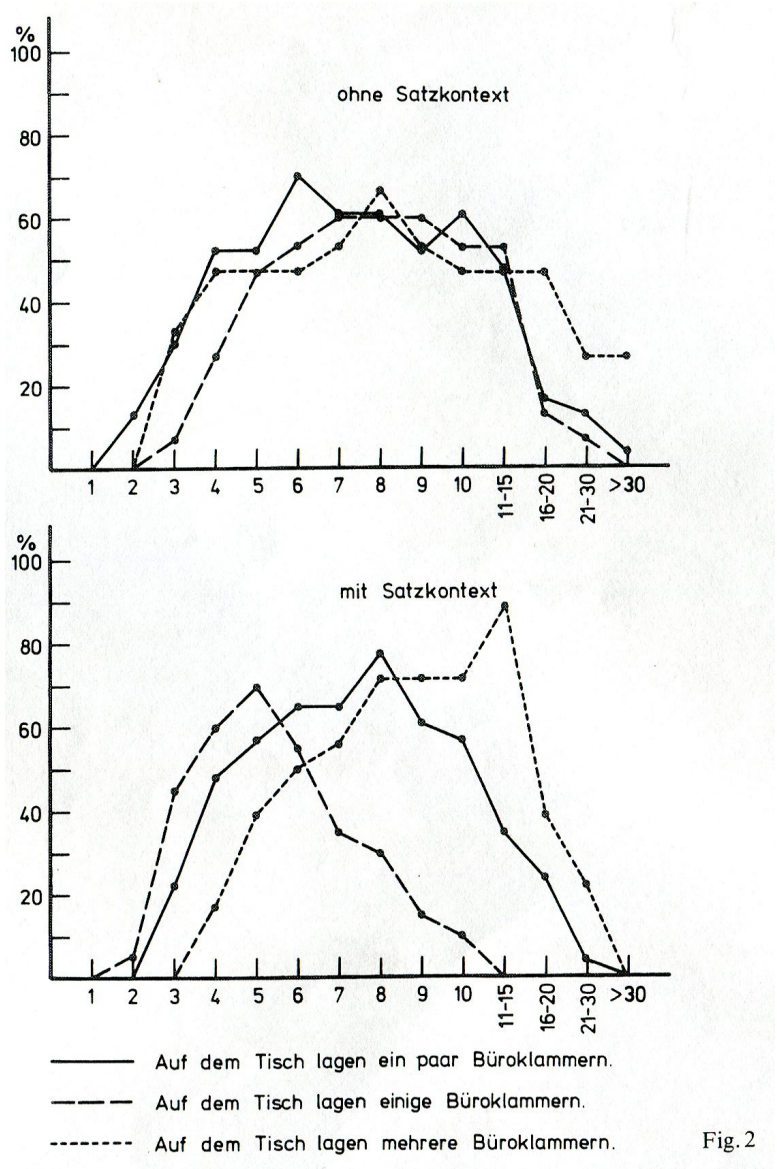



Figure 2.7: Results from Hormann (1983). Without context there is no difference between different gradable quantifiers.

### 2.4.2 Perceptual Features

Expected frequency is only one of the contextual effects that influence the interpretation of quantifiers. Since ANS is strongly influenced by spatial variations and gradable quantifiers rely on ANS, we would logically expect the judgement of gradable quantifiers being influenced by these spatial variations. There are indeed a number of reports that show such spatial effects.

Hormann (1983) tries to show that the size of the individual objects being quantified over influences their interpretation. He asked participants how many cars they would expect after hearing the phrase "Vor dem Haus standen mehrere [Autos/große Autos]" ("In front of the house there were several [cars/big cars]"). Figure 2.8 shows the results. It shows that participants systematically reported a higher number for *cars* than for *big cars*. So for the smaller object (*car*) there must be more of them to be judged *several*.

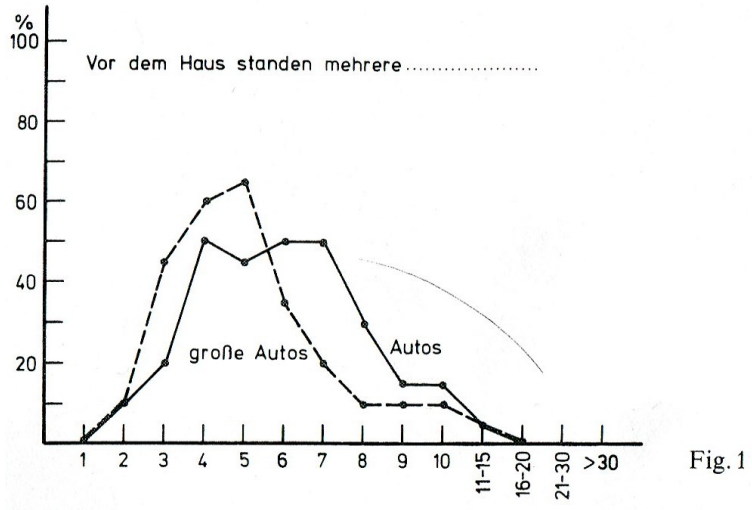



Figure 2.8: Results from Hormann (1983).

Although not disagreeing with the results, Newstead and Coventry (2000) argue against Horman's methodology. The result can also be explained by expected frequencies as described by Moxey and Sanford (1993b). For example, since *big cars* is a subset of *cars* we expect there to be less of them. In order to control for these effects, Newstead and Coventry (2000) conducted a similar experiment, but with abstract visual stimuli. They used bowls containing 3 to 21 balls. The sizes of these balls in the bowl were either all 5mm or all 12mm. Participants were asked to rate the appropriateness of the statement "There are QUANT balls in the bowl" on scale from 1 (not at all appropriate) to 7 (totally appropriate), where QUANT was one of the quantifiers *a few*, *few*, *several*, *many* or *lots*. It is shown that for the bigger balls smaller amounts are required to deem

the quantifier appropriate. The results are shown in Figure 2.9.

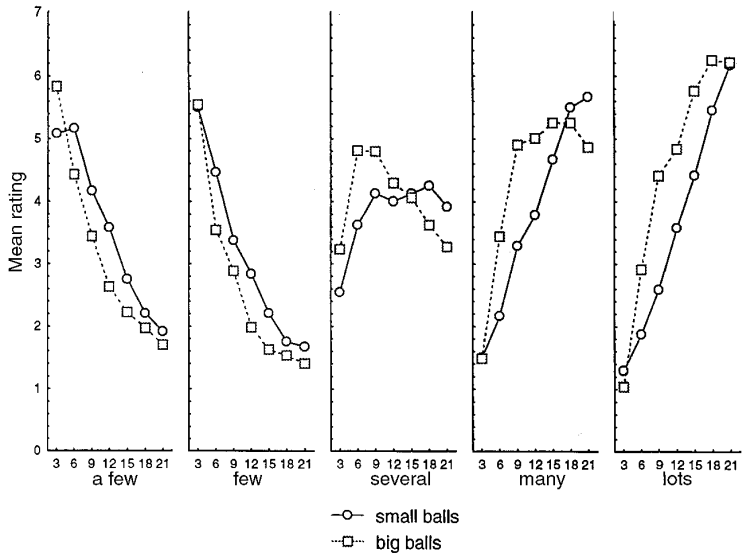



Figure 2.9: Results from Newstead and Coventry (2000).

Though the authors do not relate their findings to findings on numerosity perception, there seems to be a clear link. Depending on how we look at the situation, these results can be explained by either the effect of extent or the effect of density on numerosity perception. If we consider the bowl as part of the figure (i.e., the bowl is an integral part of the object being quantified), then the bowl with bigger objects is more densely populated. If, on the other hand, the bowl is considered part of the ground (i.e., the bowl is ignored as a part of the group that is being quantified over), then the group of bigger balls is perceived as a bigger entity. Either way, studies in numerosity perception predict that the bowl with the bigger balls is perceived as more numerous. Therefore, we would expect that the bigger balls require lower (objective) quantities to be judged appropriate for a particular quantifier. Which is precisely what Hormann (1983) and Newstead and Coventry (2000) show.

An even more direct link between numerosity estimation and quantifier judgment is found in Coventry et al. (2005). They show that the surface area of a set influences the appropriateness of quantifiers. They presented participants with images of fish and asked them to judge the appropriateness of the utterance "There are QUANT fish", where QUANT is one of the quantifiers *a few*, *few*, *several*, *many*, or *lots of*. For larger quantities ($> 12$ fish), they found that when objects are scattered over a larger area, quantifiers require smaller amounts of fish to be judged appropriate. It seems very plausible that this is the result of the effect of area size on perceived numerosity, as discussed in the first section of

this chapter (Bevan et al., 1963; Krueger, 1972). The fact that for lower quanties there is no effect is not addressed by the authors, but perhaps can be attributed to the same floor effect as suggested by Moxey and Sanford (1993b).

Experiments also show that quantifier judgment is not as clean-cut as one might think at this point. Many other subtle effects on quantifier judgement can be found. For example, the number of contrasting objects (Coventry et al., 2010), grouping of objects (Coventry et al., 2005), and attitude (Goocher, 1965) all influence the way these gradable quantifiers are interpreted.

## 2.5 Conclusion

This chapter describes research from three normally unrelated fields. To what extent the authors in different fields are aware of the work from other fields is unclear to me (they do not refer to each other), but all of them independently arrive at similar conclusions: Work in psychophysics has shown that estimation of numerosity is highly sensitive to spatial variation. Big effects have been found of variations in extent and density on the numerosity perception. Work in developmental psychology shows that numerority is a multi-dimensional concept that requires children to learn to coordinate the dimensions of extent and density. The interpretation of gradable quantifiers relies strongly on numerosity perception. So, we would expect that spatial variation also influences the interpretation of gradable quantifiers. Work in psycholinguistics shows that this is indeed the case.

The fact that different fields with different research agendas arrive at very similar conclusions only reinforces the general conclusion: Numerosity perception (and by extension the interpretation of gradable quantifiers) requires the coordination of the spatial features extent and density.

# Part II

# Methodology and Technical Background

# Chapter 3

# Planning What To Say Next: Grounded Language Processing

This chapter describes a fully operational language interpretation system that is created for robotic communicative interactions called Incremental Recruitment Language (IRL). Meaning in IRL is represented as a network (program) that contains instructions (cognitive operations). These operations can be seen as a set of instructions that allow an agent to fulfill a specific communicative goal. IRL contains a number of mechanisms to define, execute, compose, compare, and repair these IRL-programs. The goal of this chapter is to provide a detailed overview of the most essential IRL mechanisms. For implemented examples, see the IRL tutorial that is shipped with the Babel Lisp-package that can be found on `http://www.emergent-languages.org/`. This article together with the tutorial aim to provide the reader with enough information to start working with IRL.

This chapter is part of an ongoing effort to keep track of the evolution of the IRL system, as discussed below. Big parts of Section 3.2 and Section 3.3 are taken directly from the following publication: Spranger, M., Pauw, S., Loetzsch, M., and Steels, L. (2012b). Open-ended Procedural Semantics. In Steels, L. and Hild, M., editors, *Language Grounding in Robots*. Springer, New York.

## 3.1 Introduction

The main purpose of IRL is to bridge the gap between the continuous data that result from sensorimotor processing and the discrete concepts of language. Although IRL is often described as a language (as the name already suggests) or a formalism (Steels and Bleys, 2005), it might be better to conceive of IRL as a set of conceptualization mechanisms that are typically needed for robotic language experiments. Most of the mechanisms are independent of the rest of the system, so the experimenter can choose to only use the part of IRL that (s)he needs.

The early development of IRL took place at the end of the nineties. A first

implementation by Luc Steels was used in experiments in grammar emergence (Steels, 2000a). A second implementation was made by Wouter Van den Broeck (Van Den Broeck, 2008). This chapter is based on a more recent implementation by Martin Loetzsch, Simon Pauw and Michael Spranger (Spranger et al., 2010a). The current implementation has already been used in language game experiments targeting various domains including color (Bleys, 2008), spatial language (Spranger, 2011), quantifiers (Pauw and Hilfery, 2012) and temporal language (Gerasymova and Spranger, 2012a) on different robotic platforms, including the Sony humanoid (Fujita et al., 2005) and the Humboldt MYON (Hild et al., 2012) robot.

This chapter describes only the most essential mechanisms that make up IRL. The next section shows how meaning in IRL can be represented using networks of cognitive operations and semantic entities. The subsequent section illustrates the execution (i.e., interpretation) of IRL networks and their automatic composition (i.e., planning). The last section discusses the relation between IRL meaning and language, showing how IRL networks can be mapped onto linguistic constructions and in the case of noisy communication channels, how IRL can help to reconstruct the intended message (i.e., flexible interpretation).

## 3.2 Meaning Representation

IRL can best be seen as a set of independent mechanisms that are useful for modeling semantics grounded in the visual attention of robotic agents. As such, it is an open-ended system in which we can implement many different semantic paradigms. However, in order to understand the working of IRL, we focus our attention on one specific semantic model, which we will illustrate using the example utterance "the red block".

The implementation of this example is based on robotic communicative interactions, called *language games* (Spranger et al., 2010b, see Figure 3.1). In this particular language game, the robot on the left takes the role of the speaker. He picks the red block in the center of Figure 3.1 as the topic of the language game. His *communicative goal* is to find an utterance that, when interpreted by the hearer (the robot on the right), draws the attention to that object. He could for example say: "the red block." The hearer in turn interprets this utterance and points at the object he thinks the speaker intended. If he points correctly the game is a success.

In order to play such a language game, the robots are provided with computational systems for perception, conceptualization and communication that link language to the sensorimotor interaction of artificial agents. The role of IRL herein is to provide a mapping between the output of the perceptual system and the conceptual structure that the language system translates into an utterance. The perceptual systems for recognizing and tracking the objects in their envi-

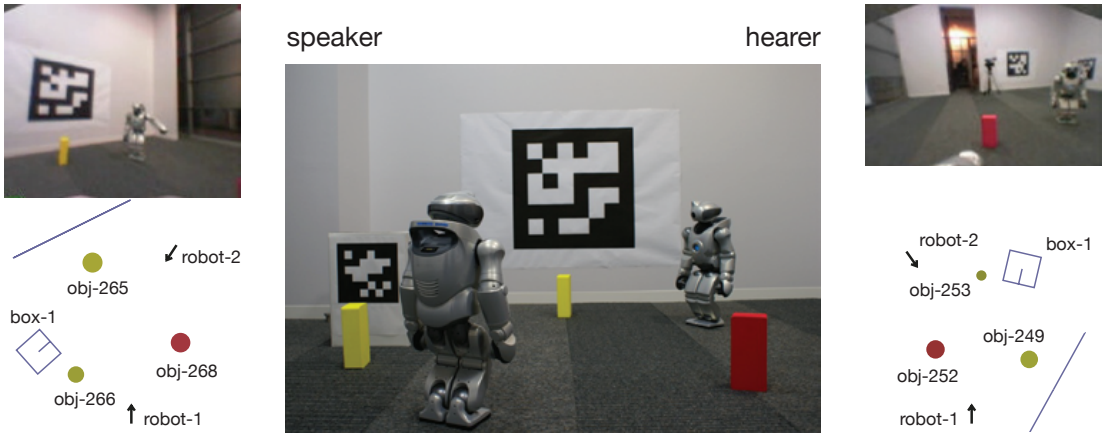



Figure 3.1: Robots scan the shared environment (center image) with the cameras in their heads (images top left and top right) and construct world models from these data streams (images bottom left and bottom right). The robot at the left has the role of the speaker and tries to draw the attention of the other robot to the red block by means of the utterance "the red block").

ronment are described in (Spranger et al., 2012a). Although these systems are not part of IRL, two important features of the output produced by these systems should be noted:

1. The world model produced by the vision system consists of a set of objects that are characterized by *continuous* real-valued features such as color, position, orientation, and dimensions. Part of the job of conceptualization is to move from these continuous values to the discrete categorizations used in language

2. The respective world models of two different robots are similar, but never identical because of perceptual noise and different perspectives. Any conceptualization mechanism used in robotic interaction experiments should be robust against this imperfect perception.

.

In order to find an appropriate description for an object, the speaker tries to find a particular set of operations that, when executed by the hearer, will single out the object from the context. Consequently, the meaning of an utterance is a set of *cognitive operations* or procedures that the speaker wants the hearer to execute in order to fulfill a communicative goal. The explicit use of operations as part of the meaning is common in *procedural semantics* (see Winograd, 1971; Johnson-Laird, 1977; Woods, 1981, for original ideas).

More specifically, an utterance encodes a *network* of cognitive operations as well as the relationships between their arguments. An example network for

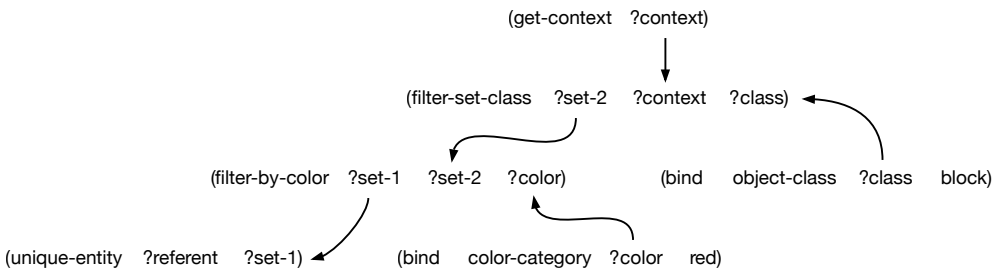



Figure 3.2: An IRL network representing the meaning of "the red block". When executed by the hearer in the interaction shown in Figure 3.1 (right robot), the variable `?referent` (the referent of the utterance) becomes bound to the object `obj-252`.

the utterance "the red block" is shown in Figure 3.2. It includes operations such as filtering the context for blocks (`filter-set-class`) or finding red objects (`filter-by-color`). Every node in the network evokes a cognitive operation, represented by its name and its list of arguments, for example `(filter-set-class ?set-2 ?contex ?class)`, evokes the `filter-set-class` operation. The arguments can be thought of as variables or slots that are bound to or will contain specific values. These slots are represented as names preceded by question marks. The same variable can re-occur in different cognitive operations, and this is represented by the arrows in the network.

These arrows are purely representational. The actual information about how the operations are linked are provided by the variable names. For example the first argument of `filter-set-class` is linked to the second argument of `filter-by-color` through the variable `?set-2`, meaning that the hearer should first filter the context for blocks and then find all red objects in this set of blocks ("red block").

There is a special operation called `bind` which introduces concrete *semantic entities* of a certain type and binds them to argument variables in the network. Semantic entities are categories in the conceptual inventory of the agents. For instance, the statement `(bind color-category ?color red)` in the example above binds the color category *red* to the variable `?color`. The color category itself has its own grounded representation.

It should be pointed out here that IRL, although designed as a procedural semantics, has some aspects that are clearly more declarative in nature. First of all, the order of the operations has no effect on the order of execution, only the relations between the operations influence their execution. Secondly, IRL operations are multidirectional. From the example above, one might conclude that cognitive operations behave like procedures as known from programming language: computing an output from input arguments. But, as we will show in in the remainder of this Section, they can operate in multiple directions, so that IRL can in fact be seen as a constraint language (as pioneered in early programming

language designs of Borning, 1981; Sussman and Steele, 1980; Steels, 1982).

### 3.2.1 Operations

The basic building blocks of IRL are *cognitive operations* and *semantic entities.* The semantic entities represent any conceptual contents—such as prototypes, predicates, relations, and sets. The cognitive operations instruct the agent what to do with these semantic entities. In this section we describe these basic building blocks in detail. The present section discusses the working of the cognitive operations and shows how one can implement such operations. The section to follow focusses on the notion of semantic entities[1].

A cognitive operation implements a specific cognitive function or task, for example filtering a set with a color category, picking elements from a set, categorizing an event, performing a spatial perspective transformation, taking the union of two sets, and so on. This is an example of how an operation for color categorization can be declared in IRL (we will show the full implementation later):

```
(defoperation filter-by-color ((target-set entity-set)
                               (source-set entity-set)
                               (color color-category))
  ;; .. implementation of the operation
  )
```

This operation has the three *arguments* `target-set`, `source-set` and `color`, which are respectively of type: `entity-set`, `entity-set` and `color-category`.

In many ways, cognitive operations behave like functions in the sense that they compute a set of output arguments from a set of input arguments. For the present example we assume that color concepts are defined by a set of prototypical colors that carve up the color space. Now, the operation `filter-by-color` can be defined such that finds all the elements in `source-set` argument of which the color falls in the color-space region defined by the `color` argument. Every object for which this is the case, is returned in the `target-set`.

However, these cognitive "operations" are not really operations, but relations; they are *multi-directional.* For example the operation `filter-by-color` can also be defined to infer a `color` category from a `target-set` of classified objects and a `source-set`. And it can be defined to compute combinations of `color` categories and resulting `target-set` values from a `source-set`. As we will show later, this ability to operate in multiple directions is crucial for flexible conceptualization and interpretation of semantic structures.

When an operation is executed, some of its arguments are *bound* to a value. This value can be any *semantic entity* (see next Section) with a type that is

[1]It should be noted that the IRL core system does not come with any built-in cognitive operations or semantic entities. The user of IRL will have to implement them for the particular experiment IRL only provides an interface for it.

compatible to the type of the argument specified in the operation. Whether an argument then is input or output of the operation depends on whether it is bound or not. To give an idea how these different cases of input-output relationships are concretely implemented in IRL, we will now show the complete implementation of the `filter-by-color` operation as used in this example:

```
(defoperation filter-by-color ((target-set entity-set)
                               (source-set entity-set)
                               (color color-category))
  ;; Case 1
  ((source-set color => target-set)
   (let ((filtered-set (apply-color-category
                        color source-set
                        (color-categories ontology))))
     (when filtered-set
       (bind (target-set 1.0 filtered-set)))))

  ;; Case 2
  ((target-set source-set => color)
   (loop for category in (all-color-categories ontology)
      when (equal-entity target-set
                         (apply-color-category
                          category source-set
                          (color-categories ontology)))
      do (bind (color 1.0 category))))

  ;; Case 3
  ((source-set => color target-set)
   (loop for category in (color-categories ontology)
      for filtered-set = (apply-color-category
                          category source-set
                          (color-categories ontology))
      when filtered-set
      do (bind (target-set 1.0 filtered-set)
               (color 1.0 category))))

  ;; Case 4
  ((target-set source-set color =>)
   (let ((filtered-set (apply-color-category
                        (color-categories ontology))))
     (equal-entity filtered-set target-set))))
```

Most of this is general Lisp code, with the IRL specific code shown underlined. Under the definition of the operation (which we have already explained before) there are four cases, which each implement the behavior of the operation for a

different combination of bound/ unbound arguments. Each case starts with a pattern that defines its applicability: when all arguments before the `=>` symbol are bound and all arguments after `=>` unbound, then the code below the pattern is executed. For example, *Case 1* specifies the operation of the primitive when `source-set` and `color` are bound, but `target-set` is still unbound.

Each case 'returns' values for all its unbound arguments with the `bind` command. For example in the first case, `(bind (target-set 1.0 filtered-set))` assigns the computed value `filtered-set` with a score of `1.0` to the argument `target-set`. In addition to that, an operation can call the `bind` command multiple times and thereby create *multiple hypotheses*. For example in the third case, the operation computes all possible pairs of values for the `color` and `target-set` arguments when only the `source-set` is bound. When multiple hypotheses are created, the scores are used to discern better from worse solutions.

It is also possible that an operation does not compute a value for an output argument. For example in the second case above, it can happen that the operation is not able to infer a `color` category which can account for a categorization of `source-set` into `target-set`. The operation will then simply not call the `bind` command, which *invalidates* the values bound to its input arguments. Finally, when all arguments of an operation are bound, then the operation does not bind any values at all but returns information on whether its arguments are *consistent*. In the fourth case, the operation checks whether the `color` category applied to the `source-set` is indeed the same as the given `target-set`.

### 3.2.2 Entities

The values that are bound to the arguments of cognitive operations are called *semantic entities*. These can be any kind of data representations, including items in the conceptual inventory of an agent (e.g. image schemata, categories, prototypes, relations, roles, etc.), representations of the current context (e.g. the world model, discourse information, etc.), and intermediate data structures that are exchanged between cognitive operations (e.g. sets of things, constructed views on a scene, etc.). In the example above, a semantic entity of type `color-category` consists of three numeric values that represent a prototypical point in the $\mathrm{YC}_b\mathrm{C}_r$ color space. The memory of the agent contains several instances of `color-category`, for example `red` is represented by the point in the color space [16, 56, 248]. Furthermore, a semantic entity of type `entity-set` represents a list of objects, which each again contain numerical values computed by the vision system. The world model of an agent is also represented as an `entity-set` so that it can be used by operations such as `filter-by-color`.

Semantic entities are typed, which makes it possible to explicitly model intuitive distinctions between different cognitive representations. Such distinctions could for example be rooted in a perceptual system which already distinguishes between objects and events because they are recognized by different sub-systems.

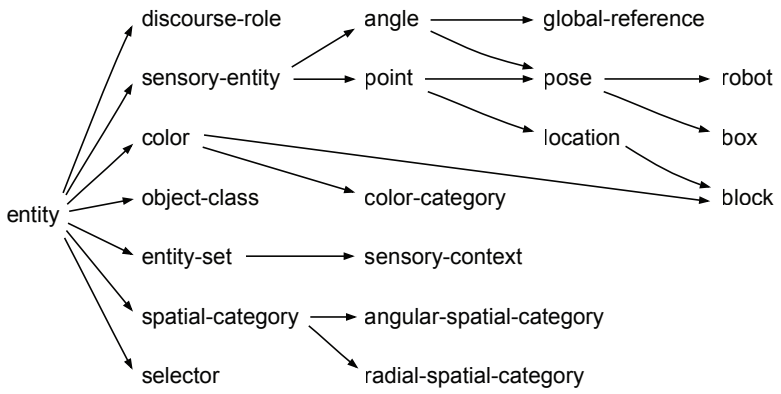



Figure 3.3: Example of a type hierarchy of semantic entities.

Or it could be the difference between a color category and a discourse role, which clearly are meant to operate in different domains. Furthermore, types can be organized in hierarchies, which allows it to treat entities with a common supertype the same. Technically, type hierarchies are represented using the standard class inheritance system of Lisp (Kiczales et al., 1991), that is new types are defined by creating classes that are directly or indirectly derived from the IRL class `entity`. Using class inheritance additionally allows it to inherit properties from other classes of semantic entities and can be used for software engineering, in particular, designs with reuse.

An example of such a type hierarchy is shown in Figure 3.3. It shows semantic entities that were chosen for the examples in this chapter. As mentioned before, the vision system used here has different mechanisms for recognizing robots, boxes and blocks. Consequently, there are different classes of semantic entities for the output of these sub-systems, namely `robot`, `box` and `block`, which all are indirectly derived from a common ancestor `sensory-entity`. In addition to that, there is the class `entity-set` for sets of objects and `discourse-role`, `color-category`, `spatial-category`, `selector` for categories that work with different kinds of cognitive operations. However, this is only an example. How concrete type hierarchies are implemented and how they interface with the rest of an agent architecture is left to the user of IRL.

Type information is used in IRL in three different ways. First, it constrains what semantic entities can be bound to arguments of cognitive operations: only entities of the same type or of a sub-type of the type of the argument or can be bound to the argument of an operation. Second, it constrains the way in which cognitive operations can be combined in networks (see Section 3.3.2). And third, they can provide a seed for semantic and syntactic categories in the grammar that expresses semantic structures: an distinction on the semantic level between objects and events could be reflected in categories such as noun and verb (see Bleys, 2008; Spranger and Steels, 2012, for experiments in this direction).

It is common practice to provide agents with an ontology that can be used to access semantical entities. Every semantic entity (every object that inherits from the object `entity`) has an identifier field (`id`). These id's are used by the semantic structure to refer to the semantic entities. In this example, the semantic entity `red`, has the id `red`. The statement `(bind color-category ?color red)` in Figure 3.2 uses this id to access that semantic entity.

### 3.2.3 Networks

Above we provided an intuitive explanation of the IRL network in Figure 3.2, representing the utterance "the red block". The network is repeated below as a Lisp S-expression. We continue with this example to explain the mechanisms for evaluation and construction of this network in more detail.

```
((unique-entity ?referent ?set-1)
 (filter-by-color ?set-1 ?set-2 ?color)
 (bind color-category ?color red)
 (filter-set-class ?set-2 ?context ?class)
 (bind object-class ?class block)
 (get-context ?context))
```

It contains four cognitive operations: `unique-entity`, `filter-by-color`, `filter--set-class` and `get-context`, and two semantic entities: `red` and `block`. The arguments of the operations are connected via *variables* (starting with a `?`). Two or more operations are linked when they share the the same variable. For example in the network above the first argument of the `filter-set-class` operation is connected to the second argument of `filter-by-color` through the variable `?set-2`.

Semantic entities are introduced in a network with *bind statements* (starting with the `bind` symbol) and they are also linked to cognitive operations through variables. For example `(bind color-category ?color red)` binds the `red` color category to the `color` argument of `filter-by-color` via the `?color` variable. The first parameter of the bind statement (here: `color-category`) declares the type of the semantic entity.

Figure 3.4 (repeated from Figure 3.2) shows the graphical representation of the network, with the links between operations and bind statements are drawn as arrows. Note that although the arrows suggest directionality, they only represent a canonical direction of execution, which nevertheless is often very different from the actual data flow in the network. Furthermore, the order of operations and bind statements in a network is not meaningful at all. All that matters is how operations and semantic entities are linked. Two networks are equivalent when both have the same set of operations and bind statements and when the structure of the links between them is the same.

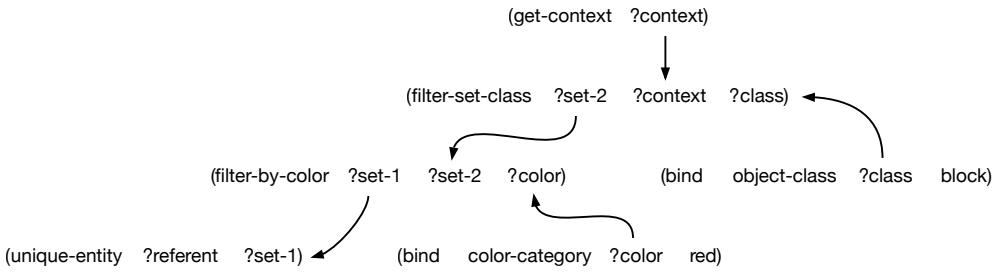



Figure 3.4: Graphical representation of an IRL network underlying "the red block". The `get-context` operation binds the set of all objects contained in the world model to the variable `?context`. Then `filter-set-class` filters this set for all objects of class `block` and binds the result to `?set-2`. This set is then filtered by `filter-by-color` for objects that match the `red` color category into `?set-1`. Finally, `unique-entity` checks whether `?set-1` contains only one object and binds the result to `?referent`.

## 3.3 Meaning Processing

The previous section shows how to define IRL-networks and all it's components. This section focuses on the mechanisms for executing and automatically composing such networks.

### 3.3.1 Interpretation

Which particular red block in the example in Figure 3.4 is referred to, i.e. which object is bound to the variable `?referent`, is found by *executing* the network within the current context of the interaction. Execution is the process by which values are bound to the variables in the network. A set of variable-value bindings is considered a *solution*, if it is *complete* and *consistent*. Complete means that all variables are bound. A set of bindings is consistent if all operations in the network have been executed.

The execution process starts by executing the bind statements to yield a list of initial bindings. The semantic entities expressed in bind statements are retrieved via their id and bound to the respective variables in the network. All other variables are assigned an empty value (unbound). As shown in the leftmost node of Figure 3.5, the initial bindings for the execution of our example network map the semantic entity `red` to the `?color` variable and `block` to `?class`, with the rest of the variables remaining unbound.

Execution of the network proceeds by executing all cognitive operations in the network. In each step, an operation is picked from the list of not yet executed operations and it is checked whether the operation can be executed given the current set of bindings for its arguments, i.e. whether it has implemented a case for that particular combination of bound and unbound arguments. If such a case

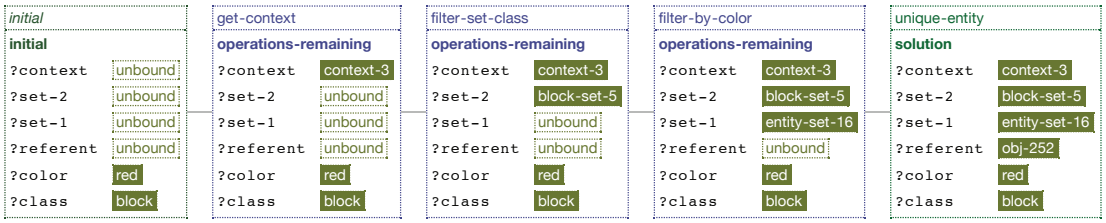



Figure 3.5: Example of an execution process. The network from Figure 3.4 is executed by the hearer in the interaction of Figure 3.1 (right robot). From left to right, each node represents a step in the execution process. From top to bottom, the executed operation, the node status, and the current list of bindings of each node are shown. A consistent solution with bindings for all variables is found in the last node, and the value `obj-252` is indeed a unique red block (compare Figure 3.1).

exists, then the operation is executed (see Section 3.2.1) and newly established bindings are added to the list of bindings. If not, then another operation is tried. A consequence of this procedure is that the particular order in which operations are executed, the control flow, can not be determined by the structure of a network alone. Rather, IRL execution is data-flow driven and execution order depends on how data spreads between cognitive operations.

In the example of Figure 3.5, the only operation that can be executed given the initial bindings is `get-context` (it doesn't require bound input arguments) and it introduces the entity set `context-3` as a value for `?context`. Then `filter-set--class` can be run, and so on. Each added binding enables the execution of more operations, until the `unique-entity` adds a binding for the last remaining unbound variable `?referent`. The set of bindings in the right-most node of Figure 3.5 is a consistent solution of the execution process, because all operations in the network have been successfully executed and all variables are bound.

Of course there can be also other outcomes of executing operations than in the example above (see Section 3.2.1). First, it can happen that an operation returns multiple hypotheses for its unbound arguments. IRL will then add each hypothesis to a copy of the current bindings list and then further process these lists in parallel. Second, when all arguments of an operation are bound, then its execution amounts to a verification or checking of consistency. If that fails, then the complete set of bindings is invalidated and not further processed. And third, when an operation is not able to bind a value for an unbound argument, then the whole bindings set is also invalidated.

To illustrate this, we will now look at the execution of a second network. It has the same operations and the same connections between them as the previous example, but does not contain bind statements for `?color` and `?class`. Instead, the `?referent` variable is bound to object `obj-268` (the red block in the world model of the speaker, see Figure 3.1):

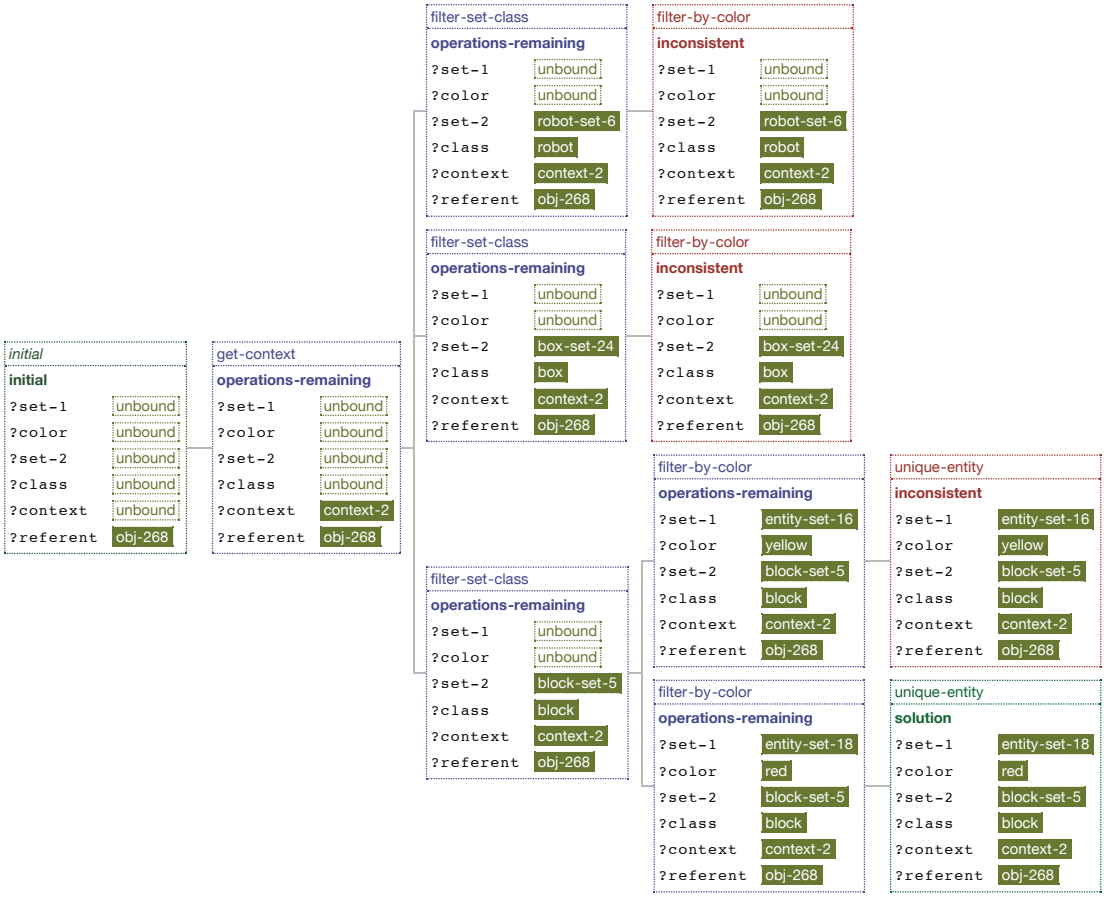



Figure 3.6: Example of an execution process with parallel processing of multiple hypotheses.

### 3.3.1. Example.

```
((bind sensory-entity ?referent obj-268)
 (unique-entity ?referent ?set-1)
 (filter-by-color ?set-1 ?set-2 ?color)
 (filter-set-class ?set-2 ?context ?class)
 (get-context ?context))
```

It is unlikely that such a semantic structure will be the result of parsing an actual utterance, but as we will see in the next section, the execution of such networks is heavily used in conceptualization. (The goal of conceptualization is to find a set of concepts that best describe a specific objects.) The execution process for this network in the world model of the speaker in Figure 3.1 is shown in Figure 3.6. Execution again starts with the `get-context` operation, but then another case of `filter-set-class` is executed: because both its `source-set` and `class` arguments are unbound, the operation creates combinations of object classes and resulting filtered sets, which leads to a branching of the execution

process. The first two of these branches (Figure 3.6 top) immediately become invalidated, because `filter-by-color` cannot apply color categories to boxes and robots. The third case, however, is further branched by `filter-by-color`, because the set `block-set-5` bound to `set-2` contains both yellow and red objects. The first of these two hypotheses is then invalidated by `unique-entity`, because `entity-set-16` contains more than one object. A consistent solution is then found with the node at the bottom right of Figure 3.6.

### 3.3.2 Conceptualization

We have seen how compositional semantics are represented and executed in IRL and will now turn to the use of these mechanisms in communicative interactions, i.e. how meanings are constructed and interpreted and how underspecified semantic structures can be completed.

For structured procedural meanings such as IRL programs, conceptualization is the process of constructing a network that, when executed, can achieve a specific communicative goal. For instance, the communicative goal can be to discriminate `obj-268` in Figure 3.1 (i.e. the red block). This goal can be achieved by the following network:

**3.3.2.** Example.

```
((unique-entity ?referent ?set-1)
 (filter-by-color ?set-1 ?set-2 ?color-prototype)
 (filter-set-class ?set-2 ?context ?class)
 (get-context ?context)
 (bind object-class ?class block)
 (bind color-category ?color-prototype red))
```

The mechanism that takes care of finding such a network is called the *composer*. The composer is implemented as a standard best first search algorithm. Starting from an initial (usually empty) network, cognitive operations are recursively added and linked until a useful network is found. Moreover, the composer can also use complete or incomplete networks in the process of composition.

An example of such a composition search process is shown in Figure 3.7. Each node in the search tree contains an (intermediate) IRL program together with a *target variable* and a set of *open variables* and a number indicating the *cost* of that node.

The target variable of the chunk in composition is the variable that is linked to the first slot of the first operation that is added by the composer (thus there is always only one target variable per network). Open variables are all other variables in the network that don't link cognitive operations. Additionally, the types of the slots of cognitive operations that are connected to target variables and open variables are also stored with the network. The cost of a node is used

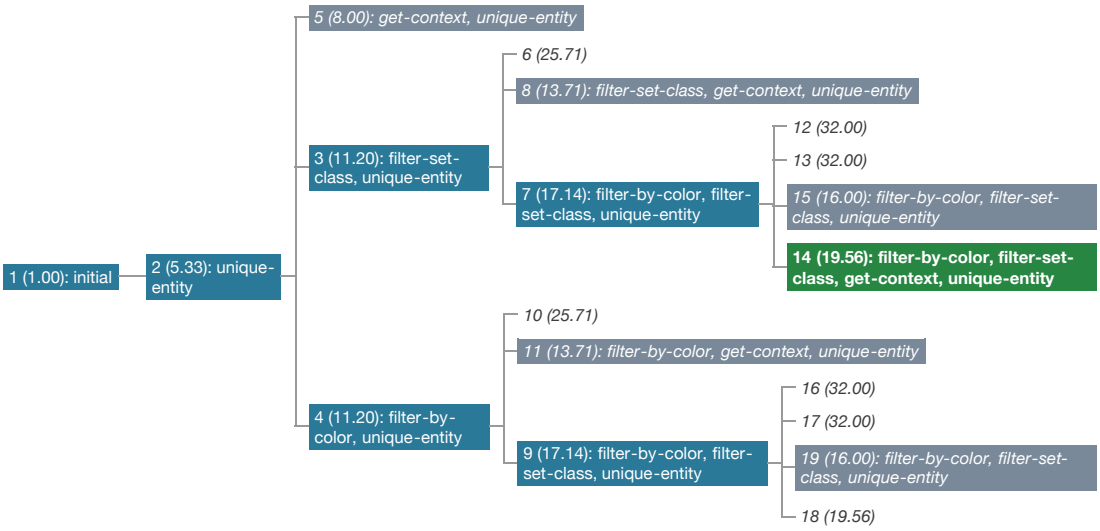



Figure 3.7: Example of a search process involved in the construction of an IRL program. Analogous to previous examples, the goal for this conceptualization process is to find a program that can identify the red block in the scene of Figure 3.1. Each node represents one processing step and branches in the tree indicate multiple possibilities for expansion. Node labels show the order in which nodes were created, a score that determines which node should be expanded next, and a list of the cognitive operations that have been incorporated into the network so far. Starting from an empty network (node 1), cognitive operations are recursively added and the resulting programs are tried out (nodes 2-3, 7, 9), until finally a solution is found that can achieve the goal of identifying the red block (node 14). By then, some nodes have not been tested yet (nodes 6,10, 12, 13, 16-18) and some can not be further expanded (nodes 5, 8, 11, 15, 19).

to determine which node to expand next. The one with the lowest cost is the first to be expanded. Example 3.3.3 shows the internal network of node '4' in Figure 3.7:

**3.3.3.** Example. Node 4

```
((unique-entity ?topic ?set-1)
 (filter-by-color ?set-1 ?set-2 ?color))
```

The target variable is `?topic` (of type `sensory-entity`) and the open variables are `?set-2` (type `object-set`) and `?color` (type `color-category`).

The search process starts from an initial node. The content of the initial node depends on the communicative goal but should always contain at least one open variable. In our example the first node contains nothing but the open variable `?topic`.

Every iteration of the search procedure consist of two phases. In a first phase the composer checks if the current networks can achieve the communicative goal.

For this, the conceptualizing speaker takes itself as a model for the hearer and executes the program using his own set of categories and his own perception of the world. This is a form of re-entrance (Steels, 2003). If one of the current networks can achieve the communicative goal then the composer is done and the solution is returned. The execution of a network can generate additional bindings. These additional bindings become part of the solution. Most of the time the networks will not provide a solution.

If no solution has been found yet, the composer tries to *extend* the network of the node with the lowest cost. The composer tries to add a cognitive operation to the existing network and links the target slot of the cognitive operation to one of the open variables of the node. This variable can only be linked if its type is compatible with the type of the target slot. For each possible extension, a child node is created with the extended network, the now connected variable is removed from the list of open variables and new open variables for the other slots of the added operation are created.

A solution of the conceptualization process is found when the execution of a node's network results in a set of bindings. The processing of nodes stops and the found program together with the bindings from execution is returned. However, often there is more then one solution and sometimes the first solution found is not the best solution. Therefore it is possible to ask the composition engine for multiple (or even all) solutions up to a certain search depth.

We will turn to an example to illustrate the expansion of a node. In Example 3.3.3, the open variable `?set-2` (of type `object-set`) of node '4' can be connected to the three different operations `filter-set-class`, `get-context` and `filter-by-color` because their target slot is of type `object-set` (which is the same as and thus compatible with the type of `?set-2`). Consequently, three child nodes are created for the three resulting networks (nodes 9-11). Node 9 contains the following expansion:

**3.3.4.** Example. Node 9

```
((unique-entity ?topic ?set-1)
 (filter-by-color ?set-1 ?set-2 ?color))
 (filter-set-class ?set-2 ?set-3 ?class))
```

Its open variables are `?color` (of type `color-category`), `?set-3` (type `object-set`) and `?class` (type `object-class`). The expansion of node '4' removed `?set-2` from the list of open variables but added `?set-3` and `?class`. This network does can not yet compute the topic. In order for the operation `filter-set-class` to compute something it requires a value for `?set-3`. But, `?set-3` is still an open variable. However, a further expansion of node 9 into node 19 does give a solution:

**3.3.5.** Example. Node 19

```
((unique-entity ?topic ?set-1)
```

```
(filter-by-color ?set-1 ?set-2 ?color))
(filter-set-class ?set-2 ?set-3 ?class))
(get-context ?set-3))
```

For the topic of this example (the red ball – `obj-268` in Figure 3.1). IRL finds an unambiguous set of bindings containing the values `red` and `block` for the variables `?color` and `?class` respectively, which was already hinted at in Example 3.3.2.

The composition process of IRL is highly customizable to the specific needs of particular learning scenarios. Most importantly, the order in which nodes are processed can be influenced by providing a function that ranks them depending on the composed program and their depth in the search tree. Nodes with a lower rank will be processed first (see the second number in the node representations in Figure 3.7). By default, networks with a low depth in the tree, few duplicate cognitive operations and a smaller number of open variables are preferred, resulting in a 'best first' search strategy. But this scoring mechanisms can also be used to implement depth-first or breadth-first searches.

## 3.4 Communication

So far we have only looked at meaning representation an processing, but we have not yet discussed how all of this relates to language. In this section we address the question how IRL-networks are translated into utterances and vice versa. IRL is independent of any specific language formalism, and there is more than one way IRL meaning could be mapped onto language. However, IRL is developed in lockstep with Fluid Construction Grammar (FCG, Steels et al., 2012a), which makes FCG the go-to candidate for parsing and producing IRL-networks. But any construction grammar would serve equally well (ECG for example, Bergen and Chang, 2005), and in many cases it would even be possible to use IRL in conjunction with categorical grammars (e.g., Steedman and Baldridge, 2009). Rather than focussing on the implementation of IRL meaning using a specific formalism, this section aims to illustrate at a conceptual level some common practices in IRL grammar design.

The second part of this section focusses on an IRL-native system called flexible interpretation. IRL is designed for robotic language games. Inherent to these language games is that communication is not always perfect. The hearer is often presented with an incomplete message—either due to transmission noise or because the hearer doesn't know all the words or syntactic constructions that the speaker uses. In such cases, the hearer can only retrieve part of the intended IRL-network. The flexible interpretation mechanism in IRL is designed to recover the missing parts of the network.

### 3.4.1 Grammar

In order to understand how IRL meaning can be mapped onto language, it is useful to point out that there are three different sources of information encoded in the semantic structure: semantic entities (bind-statements), cognitive operations, and the links between operations/bind-statements. Typically, these different sources of information all have their own particular role to play in the description of the grammar: Bind-statements are expressed by content words. For example, the word *block* maps onto the bind-statement `(bind object-class ?class block)`. Cognitive operations and the variable links are expressed by grammatical constructions. For example, the operation `(filter-set-class ?set-2 ?context ?class)` is associated with the grammatical class CN.

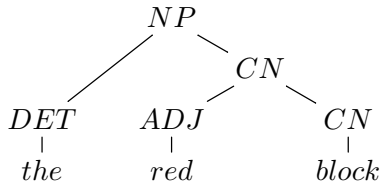



Figure 3.8: Syntactic structure of "the red block". The example grammar contains four grammatical categories: common noun (CN, e.g., block), adjective (ADJ, e.g., red), determiner (DET, e.g., the) and noun phrase (NP, e.g., "the red block"). The grammar also contains two rules for contracting utterance: The CN→ADJ/CN rule that constructs complex common nouns out by adding chaining adjectives to a simple common noun (e.g., "red block"), and the NP→DET/CN rule that constructs noun phrases by adding a determiner to a common noun (e.g., "the red block").

The syntactical side of the grammar that produces the utterance "the red block" can be represented as a rewrite grammar containing rules to produce complex common nouns and noun phrases. Figure 3.8 shows an example of the tree structure that this grammar produces. The grammar contains a rule (CN→ADJ/CN) that combines the ADJ *red* and the CN *block* into the complex CN "red block". Another rule (NP→DET/CN) adds the DET *the* to the common noun to create the NP "the red block".

The question remains: How does such a grammar map onto IRL-networks? Traditional categorical grammars would link the grammatical categories to lambda expressions that when combined will produce the desired meaning. Although it would not be impossible to create IRL-networks this way, practice shows that categorical grammars are too restrictive. Due to their more flexible nature, construction grammars turn out to be more practical for this purpose. The essence

of construction grammars is the direct link between any linguistic construction, be it a word order constraint, a lexical item, a marker or any other syntactic construction, and the meaning. Any linguistic construction maps onto part of the semantic content.

Figure 3.9 shows the mapping between syntax and meaning for the NP "the red block". In the case of content words this mapping is fairly straightforward. The word *block* maps onto the bind-statement that introduces the *block* prototype `(bind object-class ?class block)`. Similarly the word *red* maps onto the bind-statement `(bind color-category ?color red)`. The grammatical categories themselves provide the instructions of how the content words would be used. For example, the fact that *block* is used as a noun introduces the part of the network that picks all the blocks from the environment `((get-context ?context) (filter-by-class ?set-2 ?context ?class))`. The grammatical constructions that create the syntactic structure of the utterance provide the links between different parts of the network. For example, CN→ADJ/CN provides the variable link (`?set-2`) that is responsible for linking the parts of the network from the CN *block* and the ADJ *red*.

This is just one example of how semantic structure in IRL can be mapped onto syntactic structure. More elaborate examples can be found in Gerasymova and Spranger (2012b) for temporal language, in Spranger and Loetzsch (2011) for spatial language and in Bleys (2008) for color. Approaches differ in their implementational details, but on a conceptual level they all follow the present outline.

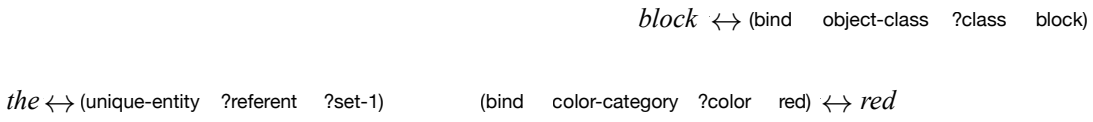
### 3.4.2 Flexible Interpretation

One advantage of using FCG over any other language formalisms to encode language is its robustness against imperfect communication. When parts of an utterance are not recognized, by the language system, FCG proceeds to parse as much of the utterance as it possibly can. So it will still recover a partial meaning. Such robustness is only helpful if the agent have some mechanism to repair this partial meaning. The presence of such a mechanism (called flexible interpretation) is one of the thinks that makes IRL such a useful system for robotic communication.

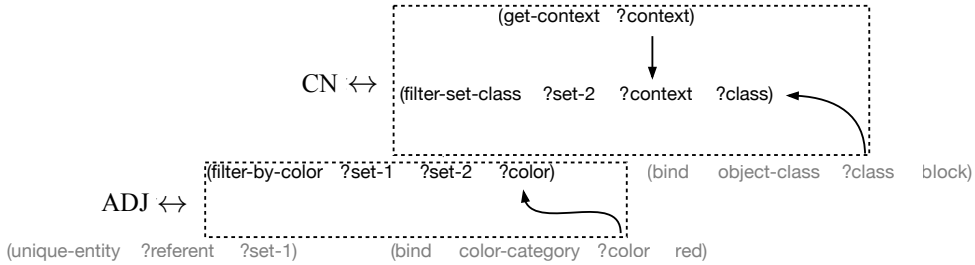
Let us consider an example. The speaker says "the yellow block right of you", however the hearer for some reason hears the following phrase:

**3.4.1.** Example. "grrgh yellow block right krkks you "

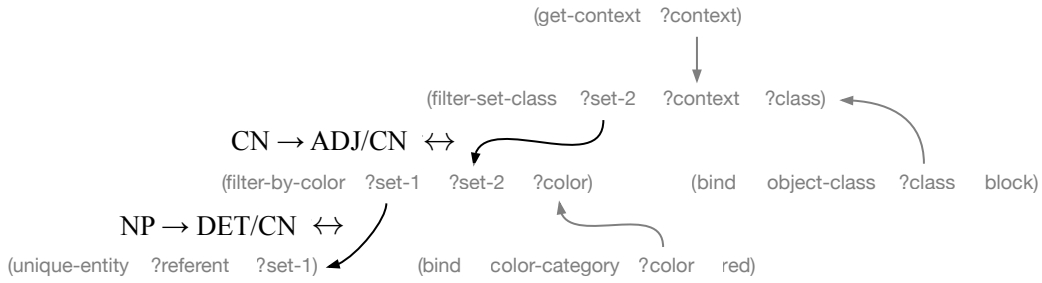
When the hearer knows English, this utterance has some recognizable elements for the hearer. Like "block", "yellow", "you" and "right", But misses "the", and "of". A language system that parses this sentence can at best retrieve only some of the intended network. Figure 3.10shows an example of a network that might be the result of parsing this utterance.




(a) Lexical Items




(b) Grammatical Categories




(c) Grammatical Constructions

Figure 3.9: The mapping between syntax and meaning. Content words provide the bind-statements (a). The grammatical categories provide the cognitive operations that link to those bind-staments (b). And the rules that that create syntactic structure provide the linking between the previously created subnetworks (c).

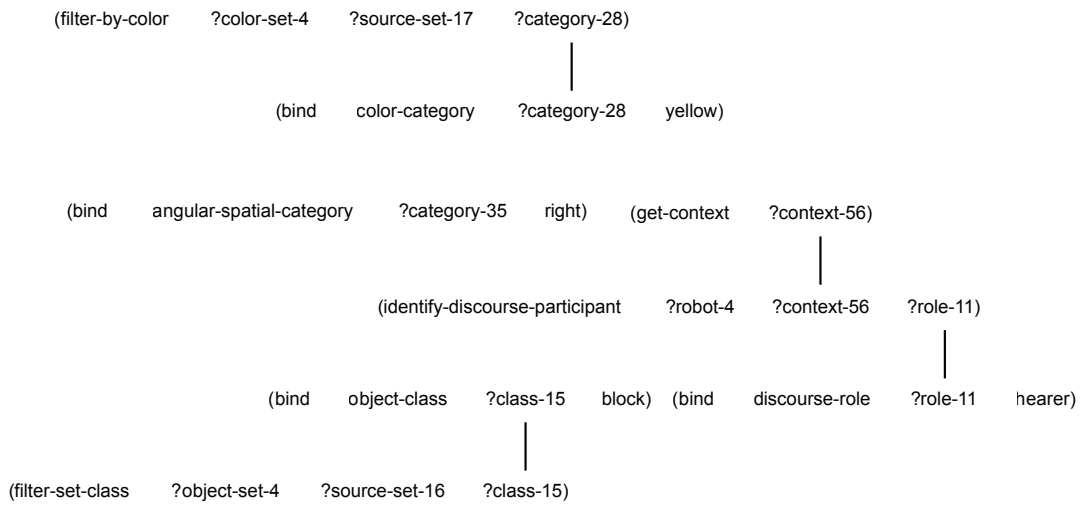



Figure 3.10: A partial network for "block yellow you".

Executing this network leads to no result (solution). However, the hearer can actively reconstruct possible meanings using the composer. The composer starts with an empty network, just as in conceptualization, and gradually adds cognitive operations building up more and more complex IRL networks. If an IRL network *matches* with the meaning obtained so far, then it is a possible interpretation of the phrase. If such the IRL network, furthermore, executes then the result of execution is considered a possible solution.

### Matching

The most important operation in interpretation is matching. The matching algorithm tests whether a particular IRL network built by the composer is compatible with the meaning that the language engine parsed from an utterance. In the next few paragraphs the word meaning is reserved for the structure parsed by the language engine (so not the reconstructed meaning). Meaning is always matched against the (intermediate) IRL networks constructed in composition.

The interpretation mechanism has to recover networks that were not properly communicated. The solution that is matched with the meaning can therefore contain additional cognitive operations and variable links. While allowing the composer to add information, the matching process has to preserve the information provided by the meaning. In other words, the IRL network that the composer finds is only a valid interpretation if it contains at least as much information as provided by the utterance.

To understand the matching process, recall that, during parsing, language constructions add meaning to the overall interpretation in the form of 1) bind statements, 2) cognitive operations, and 3) variable links. A solution in interpretation found by the composer can include additional information, but must

preserve these three aspects from the parsed meaning. Consequently, the composer has to find a network that contains at least the cognitive operations and the variable links of the meaning. In addition, the open variables have to match the bind statements of the meaning. These intuitions are captured by the following definition:

> A meaning $n$ **trivially matches** an IRL network $c$ iff (1) for each bind statement `(bind type ?variable entity)` in $n$ there is a open variable `(?variable . type)` in $c$ and (2) every primitive $p$ in $n$ is in $c$.
>
> A meaning $n$ **matches** IRL network $c$ iff there is a function $f$ from the variables in $n$ to the variables in $c$ such that $n' = f(n)$ trivially matches $c$.

where $f(n)$ is the meaning $n'$ that is the result of substituting every variable $x$ in $n$ for $f(x)$.

For example, the parsed meaning for the utterance "block" is `(bind object--class ?class block)`. This matches Network 3.4.2, but not Network 3.4.3. This is because the open variable `?class` in Network 3.4.2 is of type `object-class` which matches the object class of the bind statement. The type of the open variable `?color` in 3.4.3 is not correct.

**3.4.2.** Example.

```
((unique-entity ?referent ?set-1)
 (filter-set-class ?set-1 ?context ?class)
 (get-context ?context))
```

**3.4.3.** Example.

```
((unique-entity ?referent ?set-1)
 (filter-by-color ?set-1 ?set-2 ?color)
 (get-context ?context))
```

As a second example, consider the utterance "the block." FCG computes the following meaning:

**3.4.4.** Example.

```
((unique-entity ?referent ?set-1)
 (filter-set-class ?set-1 ?context ?class)
 (bind object-class ?class block)
 (get-context ?context))
```

This meaning matches Network 3.4.5, but it does not match 3.4.6 and 3.4.7. In Network 3.4.5, all the primitives and variable bindings of the meaning are also in the network, and the open variable `?class` matches the bind statement `(bind object-class ?class block)` from the meaning. Network 3.4.6 does not match because the primitive `filter-set-class` is present in the meaning but not in the network. Network 3.4.7 does not match because it does not have the variable link between `filter-set-class` and `get-context`.

**3.4.5.** Example.

```
((unique-entity ?referent ?set-1)
 (filter-set-class ?set-1 ?context ?class)
 (get-context ?context))
```

**3.4.6.** Example.

```
((unique-entity ?referent ?context)
 (get-context ?context))
```

**3.4.7.** Example.

```
((unique-entity ?referent ?set-1)
 (filter-set-class ?set-1 ?set-2 ?class)
 (filter-by-color ?set-2 ?context ?color)
 (get-context ?context))
```

## 3.5 Discussion

IRL has been built to support the embodied, multi-agent experiments in language evolution. This chapter discusses the mechanisms provided for autonomous conceptualization and interpretation. Namely, a mechanism for the evaluation (or execution) and composition of semantic structure, and a mechanism to reconstruct incomplete semantic structure. IRL, thus, provides the needed connection between the sensorimotor systems and the language systems, at the same time allowing for learning and open-ended adaptation.

As suggested earlier in this chapter, IRL can be seen as a *procedural semantics*. The semantic building blocks are procedures that conduct the hearer into achieving a communicative goal. And, although IRL is a novel approach to language modeling, the notion of *procedural semantics* is not. The notion of procedural semantics dates back to the seventies, to the first attempts to make computers understand human language (Winograd, 1971). The program sparked some heated debate Johnson-Laird (1978); Fodor (1978, 1979); Winograd (1975), but also culminated in a number of fruitful practical endeavours with actual systems being build. One example of such a complete system for parsing and interpreting natural language utterances is SHRDLU Winograd (1971); Hewitt (1969).

The main qualm its opponents have with procedural semantics is the lack of formal rigor (Fodor, 1978, 1979). More recent approaches in relevance theory, addressed this issue and provide a more formal version of procedural semantics (Blakemore, 1992; Wilson and Sperber, 1993; Blakemore, 2002). But, these approaches loose a lot of the procedural force. They consider procedures to be guiding the inference process of otherwise propositional meaning. Procedures, in this view are mostly an afterthought. IRL is in that sense much more reminiscent of SHRDLU then of contemporary approaches. And indeed this come at the

cost of less formal rigor. I think an important future The only argument we can make in defense simply not required for the purpose of IRL. IRL is not meant as theory, it is not designed to make predictions about language and language use. IRL is meant as a collection of mechanism that are useful for conducting grounded language experiment. And this it does well.

# Chapter 4

# Grounded Categorization and Perceptual Deviation

Grounding language in sensorimotor spaces is an important and difficult task. In order for robots to be able to interpret and produce utterances about the real world, they have to link symbolic information to continuous perceptual spaces. This requires dealing with inherent vagueness, noise and differences in perspective in the perception of the real world. This chapter presents two case studies for spatial language and quantification that show how cognitive operations – the building blocks of grounded procedural semantics – can be efficiently grounded in sensorimotor spaces.

This chapter is a shortened version of a paper by Michael Spranger and myself on this subject: Spranger, M. and Pauw, S. (2012). Dealing with perceptual deviation: Vague semantics for spatial language and quantification. In Steels, L. and Hild, M., editors, *Language Grounding in Robots*, pages 173–192. Springer, New York

## 4.1 Introduction

Noisy sensor readings and algorithmic estimation errors make it difficult for autonomous systems to acquire stable, precise, and correct estimates of the environment. Moreover, language always happens between different individuals. When two interlocutors interact in a spatial scene, they will each see the world from their viewpoint and, consequently, estimate properties of objects in the world differently. We subsume such problems under the term *perceptual deviation* which denotes that two artificial agents in the same physical space estimate the properties of objects in their environment differently.

The problem of perceptual deviation is one that humans navigating the physical world face as well. For instance, people systematically estimate distance wrongly (Foley, 1980). Humans also have vastly varying sensor precision which

has been observed, for instance, in color vision. Even people with average color vision, i.e. non color-blind subjects, have different retinal distribution of mid and long wave-length cones (Roorda and Williams, 1999). Lastly, humans interacting in spatial environments also perceive the world from their respective viewpoints.

Nevertheless, humans link symbolic information to noisy sensory information effortlessly. How this can be achieved for artificial systems has long been ignored. Traditional logic-based approaches to semantics focus almost entirely on the symbolic level and leave details of how to link semantics to sensorimotor spaces open. At the heart of such approaches is the notion of *strict* membership. A phrase such as "left blocks" is true for all objects which are blocks and to the left, in other words, all objects which are a member of the set of blocks and a member of the set of left objects. Consequently, each object in the world is either part of these sets or not. This idea can cause problems when being exposed to real-world problems such as perceptual deviation. An object might be to the left for one interlocutor but not to the left for another.

The classical approach has been criticized by psychologist and linguists alike. Rosch and Lloyd (1978), Lakoff (1987) and Langacker (1987) are examples of researchers who argue that human categorization is graded rather than strict. In their view, objects are more or less prototypical for a concept. Some objects are more `block` than others. They conclude that concepts are represented by prototypes, i.e., prototypical objects which allows other objects to be compared to them. Such a *lenient* view on the meaning of concepts has been used successfully to ground lexical language in sensorimotor streams (see Steels and Spranger, 2008; Bleys et al., 2009, for examples from action language and color). However, compositional semantics, the problem of how lexical items are combined into larger compositional semantic structures, has been mostly absent from these discussions. Furthermore, many of these proposals do not go far enough and fall back onto some version of strict membership. This chapter introduces a particularly strong version of lenient categorization that is exceptionally successful in dealing with problems of perceptual deviation and that is implemented in a larger framework for handling compositional semantics

To compare our proposal to traditional approaches, we operationalize the different ideas for a concrete piece of natural language: spatial language. We implemented spatial semantic primitives such as spatial categorization, perspective reversal and landmark processing, as well as quantifiers separately for the strict and the lenient approach in a formalism called Incremental Recruitment Language (IRL) (see Chapter 3 or Spranger et al., 2012b). We test each implementation in robot-robot interactions, called *spatial language games* (Spranger, 2011; Steels, 2012b) in which one robot is trying to draw attention to an object in the environment using spatial language. Subsequently, we can measure and quantify the success of these interactions and show why the lenient approach outperforms the classical approach.

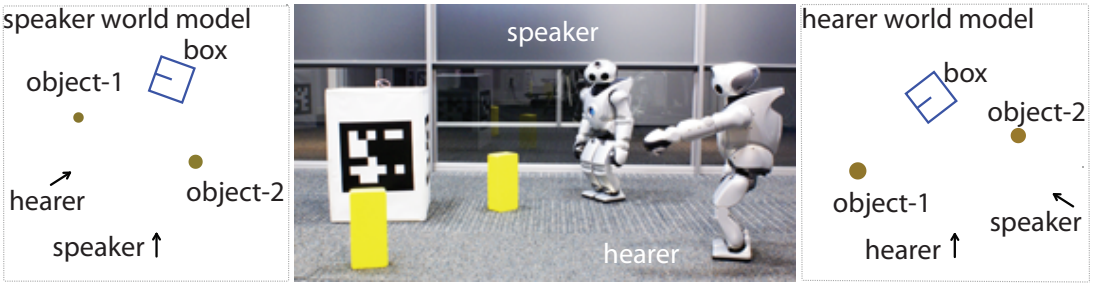



Figure 4.1: Experimental setup involving robots, blocks and a box.

### 4.1.1 Spatial Language Games

In order to study the effect of perceptual deviation we use an experimental setup in which two humanoid robots interact in a shared environment (spatial scene). One robot, the speaker, is trying to draw attention to an object in the environment using spatial language (see Figure 4.1). Here is the language game the robots play.

1. Both agents perceive the environment using their own camera. The vision system (Spranger et al., 2012a) computes a situation model (see Figure 4.1, left and right) which is comprised of blocks (circles), boxes (rectangle) and other robots (arrows). The perceiving robot is always the center of the coordinate system which is used to estimate distance and orientation of objects. The boxes have an inherent front which is visually marked and know by both robots.

2. The speaker picks an object from the context and conceptualizes a meaning for discriminating it. If he succeeds in finding an appropriate meaning, the structure is encoded in an utterance and passed to the hearer.

3. The hearer interprets the utterance by recovering the semantic structure and trying to find the object that the utterance refers to.

4. The hearer points to the object and the speaker confirms whether he pointed to the correct object.

Figure 4.1 shows real-world perceptual deviation problems. For the speaker, `object-1` is more to the left of the box (from the perspective of the box), whereas for the hearer the same object is more in front of the box (the front of the box is denoted by the small line in the rectangle). Figure 4.1 shows one scene[1] from close to 900 spatial scenes which we have recorded. Scenes differs in the number of objects and whether boxes are present or not. Some scenes have one box, some

[1]Each scene consists of two situation models, one for each robot.

| feature/measure | average | stddev | min | max |
|---|---|---|---|---|
| distance deviation | 7.2cm | 6 | 0.002cm | 59.4cm |
| angle deviation | 8° | 0.13 | 0.04° | 51° |

Table 4.1: Average, standard deviation (stddev), min and max values of angle and distance differences (angles in degrees) over 800 real-world spatial scenes.

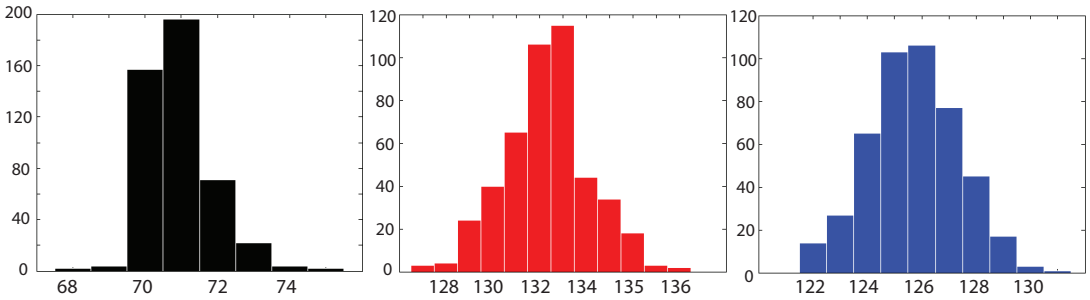



Figure 4.2: Camera noise histogram. The YCrCb values of a single pixel over time are recorded and analyzed using histograms (left - y-channel, middle - Cr-channel, right - Cb-channel

do not have any boxes. Some scenes have two objects, others up to 10 objects. For such scenes, we can precisely quantify the degree of perceptual deviation by measuring the differences in distances and angle for each perceiving robot. For example, `object-1` in the speaker world model (left image) has a distance of 81cm to the speaker. The hearer estimates the distance of the object to the speaker as being approx. 75cm. The estimation of the hearer is based on the distance he thinks the speaker has to the object. The following table shows the average differences in distance and angle measured for each robot and each object over 897 spatial scenes.

The table shows that the average perceptual deviation for the distance channel is 7cm with outliers that diverge up to 60cm. These are high numbers, but it is the sort of distance estimation problem one gets even with sophisticated computer vision systems. On average, angles diverge by around 8° with some going up to 51°. Based on these values we can conclude that perceptual deviation is always present and in some cases a quite severe problem.

### 4.1.2 Sources of Perceptual Deviation

There are four main sources for perceptual deviation: 1) sensor deviation 2) noisy or faulty sensors, 3) errors arising from algorithms used in estimating object properties, 4) differences in viewpoint on the scene.

**Sensor deviation** Sensors vary across individuals. We have already given an example for human color vision earlier. The same also holds for robots. For instance, CCD cameras from the same manufacturer have differences in light collection stats due to manufacturing margins.

**Sensor noise** Every sensor is noisy. Based on the type of sensor different sources of noise can be identified. For instance, CCD devices suffer from transfer inefficiency and shot noise (Healey and Kondepudy, 1994). Figure 4.2 shows color sensor readings taken by a digital camera in a static spatial scene. The graph shows the histograms of sensor readings from a single pixel for three different color sensors (brightness, red and blue channel). The histograms show that color readings vary over time.

**Estimation errors** Another source of errors and noise is related to algorithms used in object recognition and object feature extraction. For instance, the algorithm for the distance estimation of objects (see Spranger et al., 2012a) has distance estimation error properties shown in Figure 4.3. To estimate the position of objects the algorithm combines noisy sensor readings and integrates them over time and across different sources of information. In the process, noise and uncertainty from different sensor sources accumulate and potentially amplify.

**Differences in perspectives** Another source for perceptual deviation comes from the fact that agents perceiving the world from different bodies necessarily have different viewpoints on the scene. On the on hand, objects can look different from different angles and light conditions might vary across the environment. On the other hand, spatial properties are inherently egocentric. I can estimate the position of an object from my viewpoint, but my distance to the object is most likely different than from another person's point of view.

## 4.2 Strict Semantics

Consider an example of spatial language that highlights the problems that perceptual deviation causes for the strict approach. Suppose two robots interact in a spatial scene such as the one in Figure 4.4. The speaker says, "the block to the left of the box", to draw attention to `object-1`. For him this is an acceptable phrase for discriminating the object[2]. After all the object is the only block in the region to the left of the box. When the hearer interprets the phrase using the same mechanism, he fails. For him the object is to the right of the box and the set of blocks to the left of the box is actually empty. Obviously, the problem stems

[2] We assume an *intrinsic* interpretation of the phrase (Tenbrink, 2007).

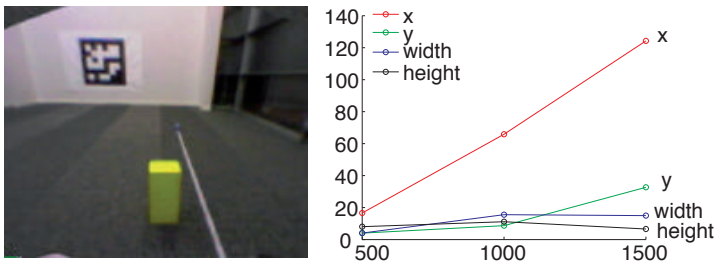



Figure 4.3: Measuring estimation errors. The block was put at 500, 1000 and 1500$mm$ distance from the robot. Each time the vision system estimates the features (width, height, x and y) of the block. The graph to the right shows the root-mean-square-error (RMS) for each measurement. The $x$ feature ($x$-axes runs towards the front) is most heavily affected by increasing distance.

from the fact that the hearer is applying a strict interpretation of the phrase. For him the region left has a fixed border and everything within the region is considered left.

Strict approaches can be implemented in different ways. For instance, a spatial relation can be characterized by the set of locations in space to which it applies (highly intractable in real-world scenarios), using regions (Kelleher and Costello, 2005), adaptive networks (Belpaeme, 2002), axioms (Eschenbach and Kulik, 1997), exemplars (Steels and Kaplan, 2002) and centroids (Bleys et al., 2009). Common to all attempts is that there are strict boundaries for category membership. An object either belongs to a certain category or not. For the sake of the argument, we only focus on centroids hereafter.

Centroids are the geometric center of convex regions in a particular sensorimotor space. For spatial relations such as `left` and `right`, for instance, centroids are points in the radial space around the robots. An object is considered to be `left` (or member of the category), when its angle is closest to the spatial point `left`, otherwise it is categorized as `right`. Consequently, every point in the sensorimotor space belongs to precisely one category from a particular set of categories and the complete sensorimotor space is decomposed into different sets of objects based on their category membership, a process known as Voronoi tesselation.

However, categorization is not enough. In order to refer to an object and try to draw attention to it, one has to *discriminate* the object from others in the set of objects. Therefore, a second condition is introduced. A category, say `left` is discriminating an object from the context, if the object is the only member of the category.

Here are the two conditions:

**Strict category membership** An object $o$ is said to be a *strict member* of the category $c$, iff $o$ is closer to $c$ than to any other category from the repertoire

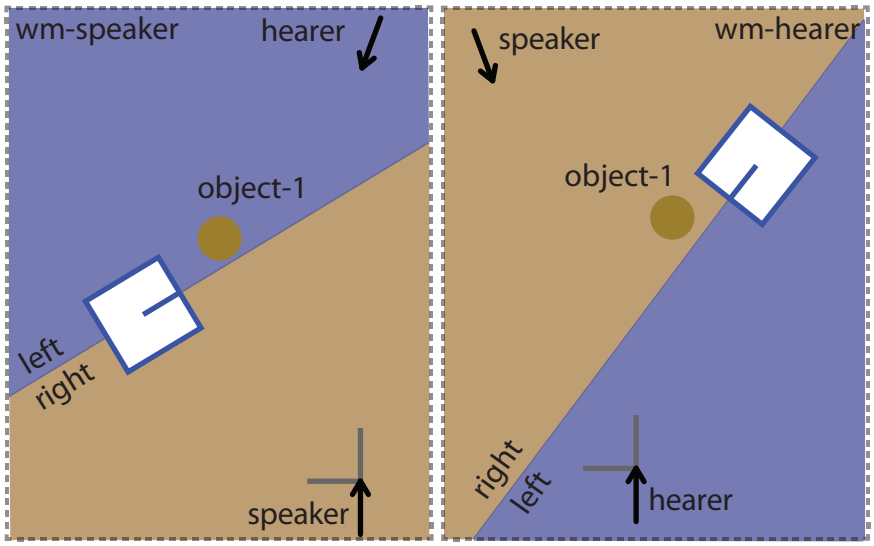



Figure 4.4: Impact of perceptual deviation. While for the robot to the left the object is left of the box, the same object is not left of the box for the robot on the right.

of categories $C$. This is known as categorization in machine learning.

**Strict discriminating category** A category $c$ is said to be *strictly discriminating* when $o$, iff $o$ is a strict member and the *only* member of the category.

Let us apply this to an example of compositional semantics say the meaning of the utterance "the left block". Figure 4.5 (left) shows the semantic representation of the utterance in IRL. The lexical items "the", "left" and "block" appear as so-called `bind`-statements which are pointers to the concept. All other nodes in the network are *cognitive operations* which denote how these concepts are processed. The referent of the phrase is computed (Figure 4.5, right) by going through every operation and executing it, a process known as *evaluation*.

`get-context` Introduces the situation model via the variable `?ctx`.

`filter-by-class` Applies the object class `block` by filtering objects in the context for those of type block, i.e. those who are strict members of the category. The result is available via the variable `?blocks`.

`filter-by-spatial-category-group-based` Applies the spatial relation `left` to further constrain the set of objects in the context for all objects that are to the left. The result is published in the variable `?left-blocks`[3].

[3]The precise implementation of this filter operation is based on the meaning of projective adverbs in English (Tenbrink and Moratz, 2003)

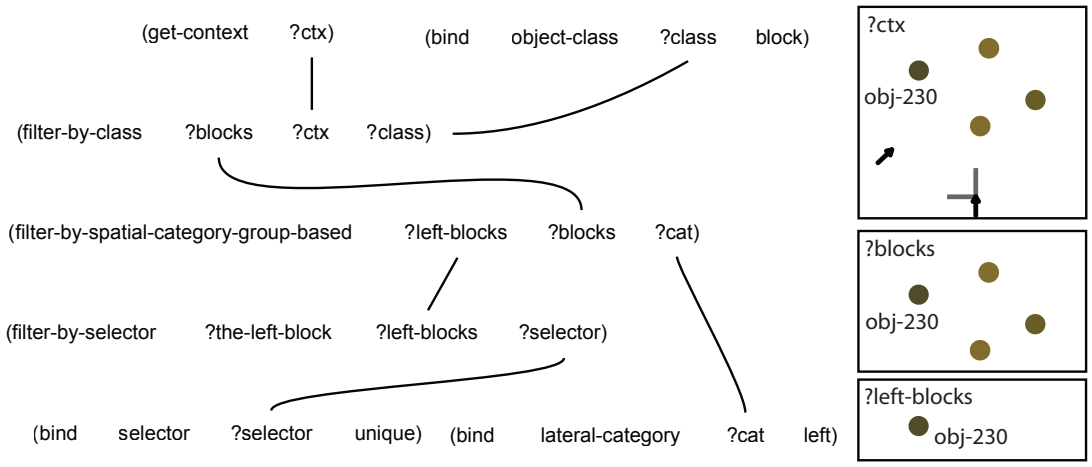



Figure 4.5: On the left side, the IRL-network of the phrase "the left block" with `filter` operations is shown. The images to the right show the progressive filtering of the set of objects in the context.

`filter-by-selector` This operation has as input the set of left blocks. It checks whether the input set contains only a *single* object (`unique`) and returns it if there is only one. This operation implements the discriminating category condition.

## 4.3 Lenient Semantics

Many scholars propose alternative principles guiding semantic processing. Rather than relying on strict membership and strict discrimination, they require that an object $o$ is the closest object to a category $c$ without further constraining the other objects in the context $O$ and their relationship to the category $c$. Consequently, other objects in the context $O$ can be strict members of the category $c$ as long as they are not closer to $c$ than $o$. Psychologists, for instance, have found that in many discrimination tasks the choice of categories seem to be based on the principle of *greatest distance* or *greatest contrast* which only requires the category to establish sufficient difference between the distance of object $o$ and all other objects in the context. These principles are used to explain human behavior in general object discrimination tasks (Hermann and Grabowski, 1976) but have also been applied to spatial language (Herskovits, 1986; Freksa, 1999). Tenbrink (2005), for instance, found that unmodified projective terms are frequently used by participants even though objects were far away from the prototypical axes.

Based on these observations, we propose a novel approach to implementing semantics which we termed *lenient*. Our approach considers similarities to cat-

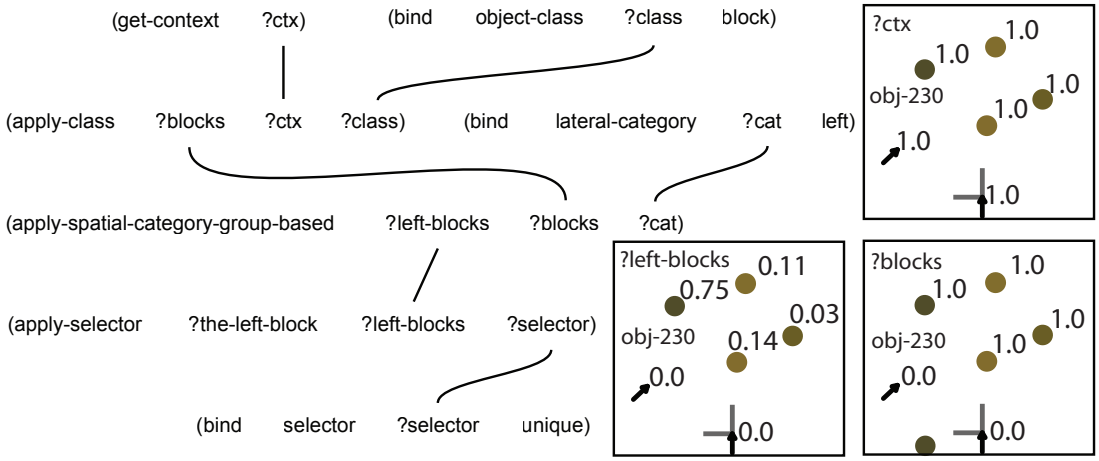



Figure 4.6: On the left side, the IRL-network of the phrase "der linke Block" (the left block) with `apply` operations is shown. The images to the right show the progressive scoring of objects in the context through the operations in the network.

egories without enforcing the strict membership criteria. Figure 4.6 shows the semantic structure for the phrase "the left block" using the lenient approach. The IRL-network is structurally the same. Only the implementation of cognitive operations is changed (signified by the prefix `apply-` instead of `filter-by-`).

`apply-class` Applies an object class by scoring each object using a similarity measure. Here, every object in the context is scored based on its similarity to the object class `block`. The result is available in `?blocks`. (Note that the membership of block is in fact strict: the similarities are either 0 or 1.)

`apply-spatial-category-group-based` Applies the spatial relation `left` by multiplying the similarity of each object with the spatial relation with the similarity to the object class `block`. The result is published in `?left-blocks`.

`apply-selector` This operation then applies the `unique` selector to the objects in `?left-blocks`. Here, this is implemented as choosing the object with the highest similarity score in the input.

The most important thing to note is that no filtering occurs. Rather, objects are scored based on their similarity to concepts and spatial categories. Only at the very end the quantifier picks the referent of the phrase. This sort of processing can deal with the initial problem presented in Figure 4.4. Upon hearing the phrase "the block to the left of the box" (see Figure 4.4), an interpreter is still able to identify `object-1` using the lenient interpretation, because the block `object-1` is the leftmost of all blocks.

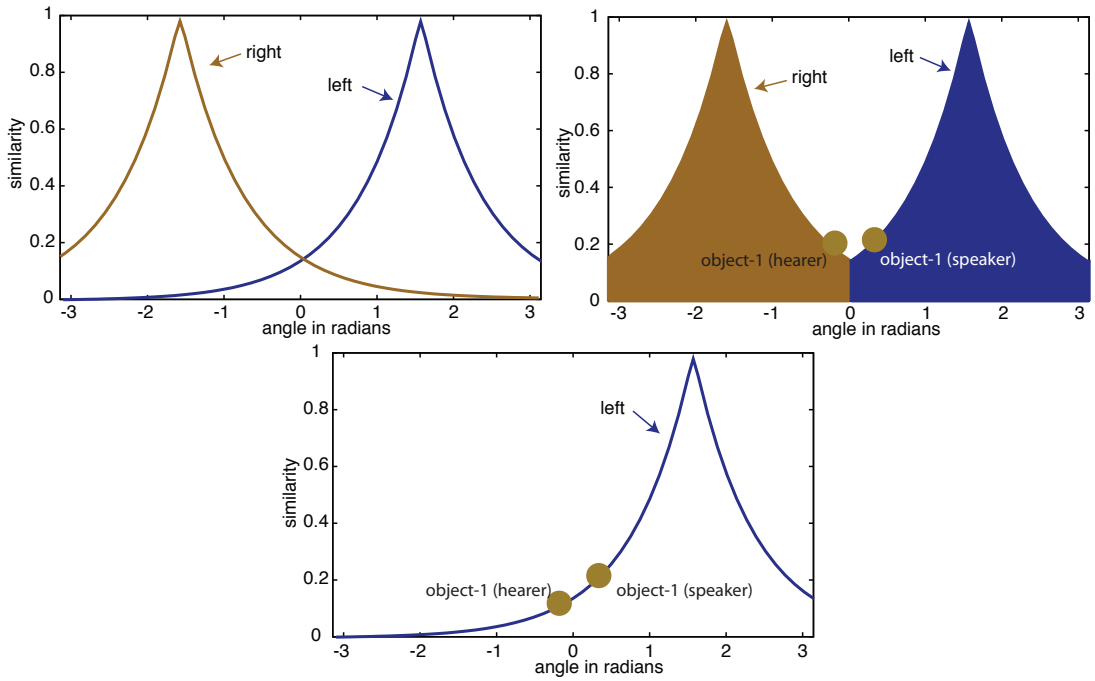



Figure 4.7: Lenient versus strict categorization. Top left figure: similarity functions for `left` and `right` categories over the angle. Top right figure: decomposition of the angular space using the strict approach. The bottom figure shows how the lenient approach uses the similarity function of the spatial category to retrieve the correct object.

Figure 4.7 shows why the lenient approach solves the situation more adequately than the strict approach. The top left figure shows the similarity functions for the spatial categories `left` and `right`. The decomposition of the angular space used in the strict approach is shown in the top right figure. The block `object-1` is categorized by the speaker as being to the left, whereas the same object for the hearer is to the right. When the speaker thus conceptualizes the object as left, the hearer has no chance of retrieving the object using strict interpretation. On the other hand, when applying a lenient discrimination scheme (bottom figure), whether or not the hearer is able to discriminate the correct object depends on whether `object-1` is the most similar object to the category `left` (which is the case, for this example).

## 4.4 Comparing Strict and Lenient Spatial Semantics

The operationalization of strict and lenient semantics allows us to study the difference between the two approaches systematically. Agents interact in controlled spatial scenes and we measure which of the two approaches performs better in a discrimination task. Here, we concentrate on the semantics only. Therefore, we scaffold syntactic processing and use *direct meaning transfer*. The hearer is passed the IRL-network conceptualized by the speaker without going through production and parsing of syntactic structure. This is equivalent to having a language without uncertainty, ambiguity or loss of information.

Section 4.1.1 describes the interaction script that is the basis of our investigation. Different steps of such an interaction can fail. We consider the following four outcomes of an interaction:

**Conceptualization failed (step 2)** After the speaker chooses a topic, he has to conceptualize an IRL-network that discriminates the topic. This process fails if the speaker cannot find any IRL-network that allows him to discriminate the object from all other objects in the context.

**Interpretation failed (step 3)** After the speaker successfully conceptualized a discriminating IRL-network, the hearer interprets this structure by simply evaluating the network. If this evaluation yields no result, the hearer is said to have failed.

**Pointing failed (step 4)** When the hearer successfully interpreted the semantic structure passed to him by the speaker, he points to the topic he interpreted. The speaker then checks whether the object pointed to is indeed the topic. If this is not the case then pointing failed.

**Success** If the hearer points to the correct object then the game is a success.

We setup two different populations of agents. In one population, agents are equipped with lenient semantics in the second population all agents are equipped with strict semantics. Both types of agents can handle the same complex spatial semantics such as group-based reference (Tenbrink and Moratz, 2003), landmarks (Mainwaring et al., 2003), frames of reference (Levinson, 1996) and perspectives (Taylor and Tversky, 1996). Agents are given a set of English proximal (near, far) (Kemmerer, 1999) and projective (front, back, left, right) spatial categories (Tenbrink, 2007). The implementation of these complex semantics is part of a larger effort on spatial language (see Spranger, 2011, for an overview).

Performance is tested on different subsets of 897 pre-recorded spatial scenes. We consider two data sets: one containing scenes with *few objects* (on average 4) and the other containing scenes with *many objects* (on average 10).

Figure 4.8 compares the lenient and the strict approach for the two sets of spatial scenes. Clearly, the lenient approach has a communicative advantage over the strict implementation. Success in interaction for the lenient approach is consistently above 85% across the two environmental conditions, whereas the success of strict categorization drops to 22% in the most difficult *many objects* condition. This means that only approx. one in four games is a success using strict interpretation compared to more than 4 out of 5 for the lenient case. Notably, the lenient approach is able to successfully conceptualize the spatial scene for the topic in question in almost all scenes. Only few cases in the *many objects* condition are marked for failure in conceptualization. On the other hand, the strict approach shows enormous problems even conceptualizing for particular objects in particular scenes. Almost all cases of failure are either due to failures of conceptualization or failures of interpretation, where conceptualization takes the major blame for failure. The two conditions show that the more objects there are in a scene the more severely the strict approach is affected.

Apart from the number of objects, the *number of categories* also influences performance. *Failures to conceptualize* are caused entirely by insufficient clustering of the input space. The problem is that there are not enough categories to allow the speaker to discriminate the topic object. On the other hand, *failures to interpret* and *pointing failures* are caused by perceptual deviation. In order to control for lack of categories, we compare four additional conditions: *english*, *double*, *triple* and *quadruple*. The *english* condition is the same as used in the previous results: agents are given sets of English categories. In the *double* condition, the number of categories is doubled. Instead of two lateral categories left and right there are now four. The same holds for frontal and proximal categories. In the triple and quadruple condition, agents are equipped with three and four times as many categories. In each condition the sensorimotor space is equally decomposed by the categories.

Figure 4.9 shows results. The left two groups of bars show the performance of the lenient approach versus the right two groups which show the strict approach. Results reveal not much change for the lenient approach. However, the perfor-

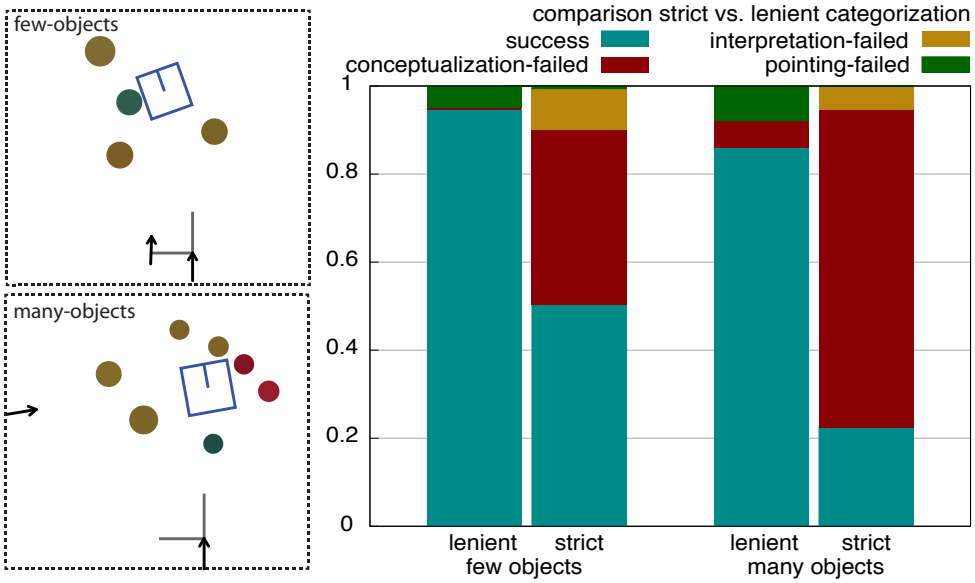



Figure 4.8: Results of comparing strict versus lenient categorization (right image) on different sets of spatial scenes (left images).

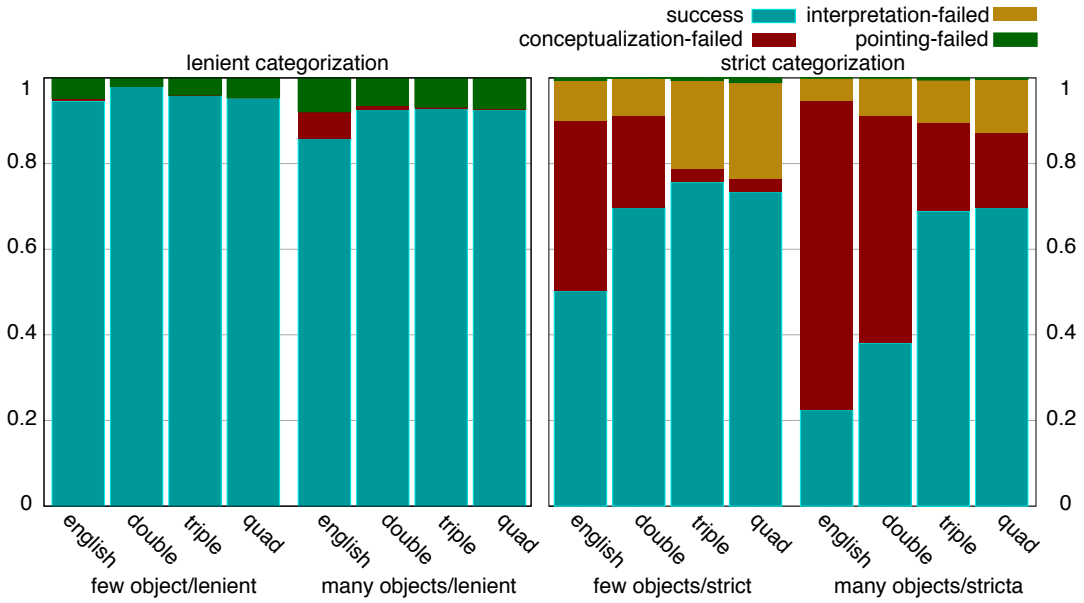



Figure 4.9: Results of comparing different sets of spatial categories and their effect on strict and lenient semantic processing.

mance of the strict approach increases drastically with more categories. But, the graph also shows a saturation effect. Success actually drops again for quadrupled number of categories. Failures of the speaker to conceptualize are replaced by the inability of hearers to interpret and, to a lesser extent, by errors in pointing. This means that the more categories there are available the more impact perceptual deviation has on the strict set approach. The reason is that the more categories, the smaller the area of categories in the sensorimotor space. Consequently it becomes more likely that an object categorized as belonging to a certain category by the speaker will be categorized differently by the hearer.

## 4.5 Discussion

Up to this point we have given a detailed account of our lenient approach to semantics in search of a solution for the problem of perceptual deviation. However, in the field of linguistic vagueness the aptness of such an approach is highly debated. Although, the problems that are being discussed in this field are of a very different nature than our own, we feel that we can not entirely omit touching upon this discussion.

In linguistics, the discussion of vagueness focusses mainly on gradable adjectives such as "tall]' or 'bold' that have no clear semantic boundaries (see van Rooij, 2011 for an overview). Gradables can be be relative ("tall") or absolute ("flat"). Precise concepts can be made vague, for example by using hedging expressions (Lakoff, 1973) such as "about" and "roughly". And, even seemingly precise concepts are often used in a vague way. For example, a round number such as "twenty" is often used as an approximate (Krifka, 2007).

The need for a model that can deal with vagueness is widely recognized, however the kind of model that should be used is a point of dispute. Traditionally, vagueness focusses on the existence of borderline cases of utterances such as "john is tall". An example is three valued logic where such an utterance can be true, false or undefined. Most modern accounts of vagueness fall under one of two approaches: Degree-based and delineation-based approaches to vagueness (van Rooij, 2011).

Degree-based approaches (Zadeh, 1965; Stechow et al., 1984; Kennedy, 2007) assume that category membership can be expressed in terms of degrees. Such a degree is typically a score between 0 and 1. The model as presented in this chapter is an example of such an approach.

Delineation approaches (Lewis, 1970; Kamp, 1975; Klein, 1980) assume that gradable adjectives are strict predicates, but that the membership of an individual is context dependent. For example, "tall" and "bald" do have cut-off points in every specific context, but the actual cut-off point for all possible contexts is underdetermined. Super-valuation is an example of such an approach.

Clearly, the model we propose is an example of a degree-based approach and is

therefore subject to the objections that come with such approaches. Proponents of delineation approaches point out two problems with degree-based analyses. First of all, degree-based approaches fail to preserve necessary logical properties. For example in Fuzzy Semantics $p \wedge \neg p$ is not necessarily false. Secondly, it is not clear what the degrees reflect or where they come from. Are they probabilities? Neuron activation levels?.

So why do we use a degree-based approach in spite of these objections? First of all, most of the problems with degree based approaches are well beyond the scope of the current model. The language games require the agents to discriminate objects to each other, not to establish truth. For this purpose the question if specific logical properties are respected is not of much concern. For example, the inference "x is taller than y" implies "y is shorter than x" is not addressed in referential language games. The second more important reason is of a more practical nature: Degrees are the vantage point of our model. The data as described above is continuous. The classification of a perceived object requires some sort of comparison to an internal representation based on similarity measures. A degree-based approach that directly operates on these similarity measures provides therefore a straightforward model of semantic processing. Lastly, delineation-based approaches are not impervious to complications either. Proponents of fuzzy logic (Lakoff, 1973; Wright, 1975; Kamp, 1981a) argue that such accounts are inadequate because they still rely on unnatural borders. It is cognitively implausible that a cut-off point for the word "tall" exists, even for one particular valuation function. And even if it does exist, the ontological status of such a valuation is just as unclear as that of the degrees. So, it is not the case that there is a problem free, ready to use alternative that we are omitting.

## 4.6 Final Remarks

Obviously, traditional approaches to semantic processing have a lot to offer and made many important contributions, for example, related to reasoning. In our view, the way to combine the two approaches and therefore leverage the great results of logic-based semantic theories is by distinguishing discrimination from description. The notion of truth makes a lot of sense in description tasks, where an accurate description either fits a situation or not. This contrasts with discrimination tasks where truth is not an immediate concern but rather the contrasting of objects from other objects seems dominant. The lenient approach works well in discrimination tasks and can easily be extended to work in description tasks, for instance, by reintroducing acceptability limits. The lenient mechanisms allow agents to track how acceptable a category is for a particular object and, hence, can also be used to make true/false distinction thresholding the similarity landscape and modulating the interpretation of quantifiers in determined noun phrases.

This chapter has argued for a particularly lenient way of grounding meaning

in sensorimotor data streams. We have taken the domain of spatial language and illustrated the practical effects and advantages of our model. We compared the performance of the lenient approach to the dominant approach in semantic theory and argued that our approach outperforms traditional semantic processing in discrimination tasks. The experiments show that real world tasks require a rethinking of deep aspects of semantic theory.

## Acknowledgements

The research reported here was carried out at the Sony Computer Science Laboratories in Paris and Tokyo. We are greatly indebted to Masahiro Fujita, Hideki Shimomura, and their team for creating the Sony humanoid robots and for making them available for the experiments reported here. This research was funded by the Sony Computer Science Laboratory in Paris with additional funding from the ECAGENTS and ALEAR projects funded by the EU FP6 and FP7 frameworks.

# Chapter 5

# Clustering Quantifiers and Perceptual Deviation

This chapter studies how quantificational expressions such as *few*, *three* and *all* can be grounded in real-world perception. Based on findings from psycholinguistics, we propose a computational model designed for use in robot-robot interaction scenarios which involve discrimination tasks for objects in the real world. We test the performance of our model and contrast it with a type theory based model. We show that our design choices make our model more suitable for real-world applications. This chapter has previously been published as Pauw, S. and Spranger, M. (2012). Embodied quantifiers. In Lassiter, D. and Slavkovik, M., editors, *New Directions in Logic, Language and Computation*, pages 52–66. Springer

## 5.1 Introduction

The experiments reported in this chapter are part of a greater research effort that studies human language-like communication using (artificial) robotic agents (Steels, 2012b). Central to these studies is the question: How can the meaning of language be grounded in real-world perception? Answers to this problem are given for different aspects of human language such as color (Bleys et al., 2009), space (Spranger, 2012), temporal language (Gerasymova and Spranger, 2012b) and action language (Steels et al., 2012b; Steels and Spranger, 2012; Spranger and Loetzsch, 2009). All of these models operationalize basic insights from prototype theory (Rosch et al., 2004) about how people conceptualize objects and relations between them and propose a degree-based semantics. In this chapter we describe a fully operational model for natural language quantifiers such as *many*, *all* and *three* that builds further on these findings. The model, termed *clustering quantification* (Spranger and Pauw, 2012), employs a combination of prototype theory and standard clustering algorithms and has been successfully used to study the acquisition and evolution of quantificational terms (Pauw and

Hilfery, 2012; Pauw, 2013b).

Inspired by existing psycholinguistic research on quantification (Hormann, 1983; Newstead and Coventry, 2000; Coventry et al., 2010), the model presented in this chapter focuses on the role of quantifiers in determining a referent of a quantified noun phrase. The quantificational information of a noun phrase imposes constraints on the cardinality of its possible referents. For example, the quantifier *three* in the utterance "the three blocks" signals that the extension of *blocks* in the context contains three elements.

We test the adequacy of the clustering quantification model for real-world perception through a series of experiments. In these experiments we contrast the performance of our model with a model that is more in line with type theoretic accounts of quantification which assume that nouns can be modeled as predicates. Accounts falling into this class are Generalized Quantifier Theory (GQ) (Barwise and Cooper, 1981) and Fuzzy Quantifier Theory (Zadeh, 1965, 1983). This chapter proceeds by introducing the embodied interaction paradigm. The section to follow introduces our model for quantification. After that, we introduce the type theory based model. Finally, we compare both models and show that the clustering quantification performs significantly better.

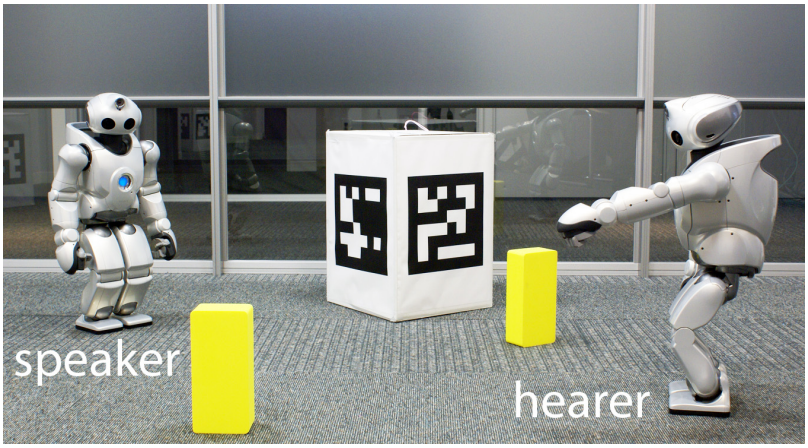
## 5.2 Embodied Interaction

The model presented in this chapter is designed for use in real-world situated interaction. Figure 5.1 shows an example scene with two Sony humanoid robots (Fujita et al., 2003) interacting in a shared environment. Each robot perceives the world through its own onboard sensors, e.g., the camera and proprioceptive sensors. The vision system (Spranger et al., 2012a) gathers information from the sensors into a *world model*, that reflects the current belief of a robot about the state of the environment. One of the robots is randomly chosen as the speaker and he will choose a referent, which can be any object or subsets of objects in the environment. The goal of the speaker is to draw the attention of the interlocutor to the referent and make him point to it or to each of the objects that are the referent.

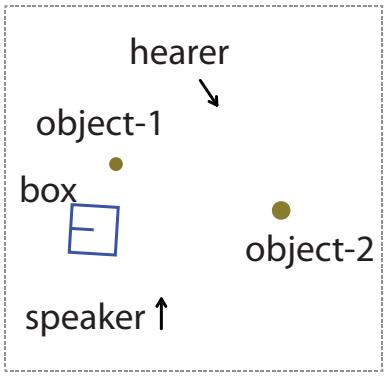
We call these interactions *language games* (Steels, 2001). The type of language game the agents play depends on the particular research question. For this chapter, the two interacting agents use the following script:

1. Both agents establish a joint attentional frame (Tomasello, 2003) and a world model using their visual and proprioceptive sensors.

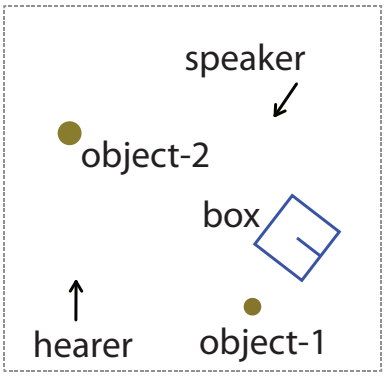
2. The speaker chooses an object or a set of objects as referent $R$. He conceptualizes a meaning for discriminating $R$ and tries to verbalize his conceptualization into a string of words.




(a) Example scene




(b) Speaker world model




(c) Hearer world model

Figure 5.1: (a): Example scene consisting of various objects, e.g., robots, blocks and boxes. (b) and (c): Top-down view of the world models as perceived by respectively the speaker and the hearer.

3. This utterance is passed to the hearer.
4. The hearer parses and interprets the utterance and tries to find the object or the set of objects he thinks the speaker is trying to discriminate.
5. The hearer points to the object or the set of objects.
6. The speaker checks whether the objects pointed to by the hearer were indeed the ones he had in mind.

This language game can have two different outcomes. If the hearer points to the correct set of objects the game is a success. Otherwise it is a failure.

Such interactions require a mapping of continuous perceptual data to discrete symbols (language). To this end, we use the computational semantics systems *Incremental Recruitment Language* (IRL) (Spranger et al., 2010a, 2012b). Since

the communicative goal is to identify some referent, the semantics of a particular phrase is modeled in IRL as a series of operations, i.e. a program, that the hearer has to go through in order to single out the objects that are the referent.

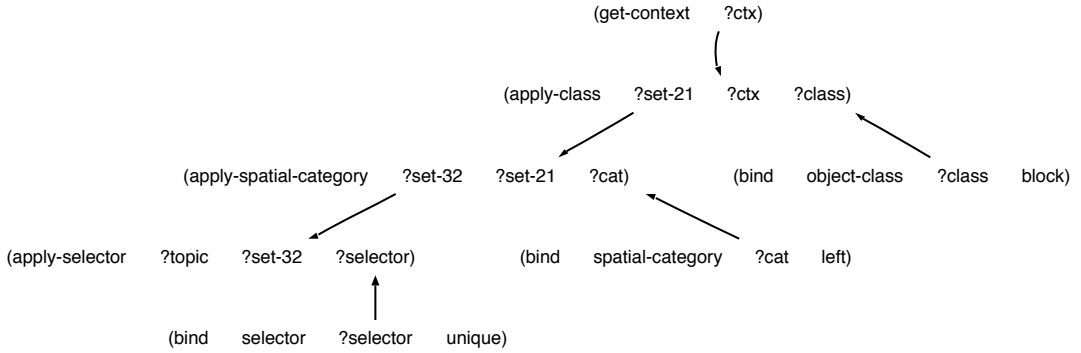



Figure 5.2: Semantic structure representing the meaning of the utterance *the left block*. The network contains bind-statements that introduce semantic entities (e.g., the object class `block`), as well as operations that define what to do with these semantic entities. Links in the network are defined by variables (starting with a `?`), e.g., the output of operation `apply-class` is linked to `apply-spatial-category` through the variable `?set-21`.

Figure 5.2 shows the IRL-program underlying the utterance *the left block*. The program is represented as a network containing *semantic entities* (e.g., `block` and `left`) and *cognitive operations* (e.g., `apply-class`). The semantics entities represent concepts and categories. The cognitive operations instruct the agents what to do with the semantic entities. For example, `apply-class` takes the concept `block` and applies this to the objects in the context. The result of this application is fed to the operation `apply-spatial-category`, which processes the data using the spatial category `left`, and finally, `apply-selector` computes the referent using the selector `unique` for more information.

IRL provides a general framework for the automatic interpretation and composition of such programs, but the concrete implementation of each operation, e.g. `apply-selector` is outside of IRL. IRL makes no assumptions about the inner workings of these operations. Consequently, IRL is an ideal formalism for studying different models of categorization and quantification.

## 5.3 Clustering Quantification

There is been substantial research in the past 10 years on grounding basic categories and relations in real world perception. We build upon an existing system for spatial language which has been proposed for the grounding of spatial categories and quantifiers such as “the” (Spranger, 2011) and has been shown to

be very successful in real world interactions (Spranger and Pauw, 2012). This system is based on two psycholinguistic processing principles.

**acceptability** (Herskovits, 1986), also called *prototypicality* (Rosch and Lloyd, 1978; Lakoff, 1987), means that categories such as *left* apply to a certain degree. An object can be more or less to the left of a landmark.

***contrast*** requires speakers which are trying to discriminate objects to choose the relation or category which maximizes acceptability of the object and minimizes acceptability of all other objects (Tenbrink, 2005). The phrase "the left block" refers to the leftmost block in a scene.

Starting from this model the main question is how notions of acceptability and contrast can be extended to quantifiers which might introduce additional constraints such as cardinality (e.g. "three"). In this section we propose to operationalize these ideas for quantifiers using mechanisms from machine learning and clustering. Before we jump to the quantifier model we briefly outline how prototype-based processing is implemented for categories.

### 5.3.1 Acceptability

The acceptability of a concept for an object depends on a similarity function that assigns a score to the combination of concept and object. For example a spatial relation such as *left* is represented by a prototypical vector in euclidean space. The degree to which an object is left depends on the angle between the object and the prototypical vector for *left*. The similarity function maps this angle difference to a score between 0 and 1. The following gives an example of a parameterized similarity function used for modeling projective categories such as *front* and *left*.

$$s = e^{\frac{0.5\mathrm{d}(p,o)}{\sigma_p}} \tag{5.1}$$

With $s$ being the resulting similarity score, $\mathrm{d}(p, o)$ a distance function between prototype $p$ and object $o$, and $\sigma_p$ a parameter that determines the rate at which the distance influences the similarity. In the case of projective categories such as *front* and *left*, this distance function computes the difference in angle between object and prototype. Similarly, functions for other concepts such `block` are constructed, except that the similarity/distance space might be the set of features of an object.

For semantic structure such as the one in Figure 5.2 this means that operations that apply categories such as `apply-spatial-category` and object classes such as `apply-class` assign scores to the objects in the context. Scores for the spatial category and the object class are multiplied so that in the end a single acceptability rating in the form of a similarity score is computed for each object in the context. In short, this is a model for spatial language which establishes

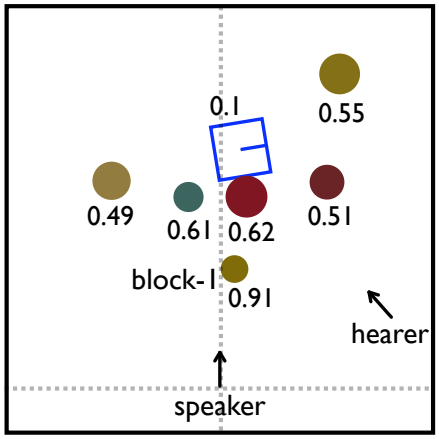



Figure 5.3: This figure shows the results of the interpretation of the utterance "block(s) in front". Every object is assigned a score that is the result of multiplying the respective score for the categories *front* and *block*.

the acceptability of a noun phrase such as "block in front of me" for every object in a given context (See Figure 5.3).

### 5.3.2 Contrast

But, how do agents use these scores to distill concrete referents? This is where the notion of contrast comes into play. For instance, if an agent wants to discriminate an object from the context then he is likely to try and maximize the applicability contrast (the difference in similarity scores) between the object he wants to discriminate and all other objects in the context. Hearers choose the object that best fits the description. Speakers choose the categories that maximize the contrast.

Similarly, if the referent is a set of objects (as for the utterance *three blocks*) we require a procedure that decides for every entity whether it is part of the referent set or not. Quantificational information constrains this process. For example, the quantifier *three* signals that the referent set contains (at least) three element, and the quantifier *many* signals that the referent set contains more elements than a certain norm. To operationalize these ideas we use standard clustering algorithms from machine learning. In particular, we apply variants of *agglomerative clustering* (Mitchell, 1997) and *k-means* (Lloyd, 1982; Manning et al., 2008).

The algorithms are used to implement the operation of `apply-selector`. The task of this operation is to decide for every entity in the context if it is part of the referent or not, based on the scores that were assigned by the previous operations. In essence it has to divide the input set into two sets of objects,

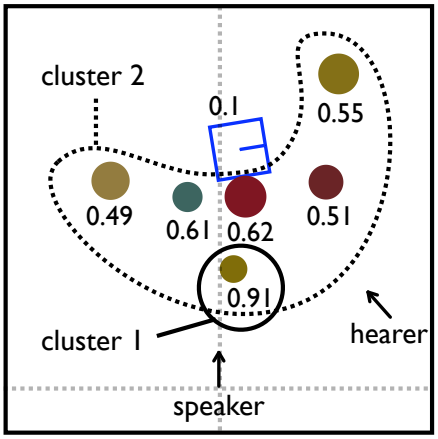



Figure 5.4: This figure shows the results of applying agglomerative clustering. The algorithm finds two possible referents for the utterance "block(s) in front" (cluster-1 and cluster-2). The quantificational information of the noun phrase can be used to further constrain the possible referents of the noun. The noun phrase "the block in front of me" signals that there is one unique referent, making cluster-1 the most likely referent. For the noun phrase "all blocks in front of me", the most likely referent is cluster-2.

the objects that are part of the referent (REFSET) and the objects that are not (COMPSET). Finding such a partitioning is precisely what clustering algorithms are designed for. However, there are many ways to partition an input set. The particular partition that is chosen depends on a combination of factors. First of all, the clustering algorithms use heuristics to find good partitioning. Good partitionings are those that maximize *inter-cluster variance* and minimize *intra-cluster variance*. The first is a measure of how far clusters are apart (contrast). The second is a measure how much cohesion there is in each cluster. Both k-means and agglomerative clustering are algorithms that optimize for these two indicators and we apply them here to similarity scores computed for the spatial relation and the object classes. Figure 5.4 shows an example result of the clustering algorithm. For the present experiment, the precise details of this clustering are not relevant. With different parameter settings different clusters could have been computed. The only constraint is that all agents use the same clustering algorithm.

Another factor that is taken into account in determining a good partitioning is the quantificational information. The quantificational information provides information on the cardinality of the REFSET. Consider for example Figure 5.4. For the utterance "all blocks in front of me", the plural marker and the quantifier *all* enforce that the REFSET should at least contain two (and preferably more) elements. The REFSET for the utterance "the block in front of me" should contain precisely one element. Thus depending on the quantificational information,

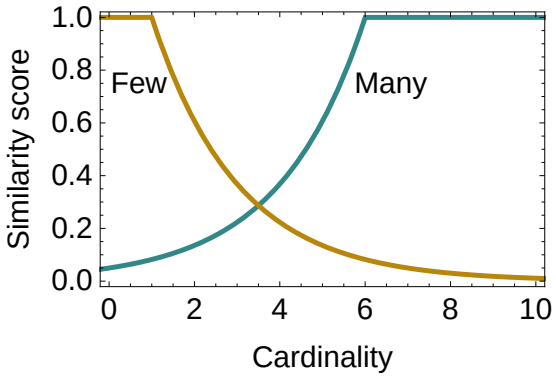



Figure 5.5: Plot of the similarity functions of *many* and *few*. The prototypical value for *few* and *many* are 1 and 6 respectively. They intersect at 3.5, meaning that for any cardinality above 3.5 the quantifier *many* is more acceptable, and for any cardinality under 3.5 the quantifier *few* is more acceptable. In practice, the similarity function is modulated by a context parameter which shifts the exact point where *few* becomes more acceptable than *many*. For the purpose of this chapter, the parameter is left fixed.

a different partitioning is chosen.

The result is a flexible algorithm which allows agents to choose a partitioning of data based on whatever quantifying criteria they want to convey. Conversely, it allows them to find the best interpretation upon hearing a quantified noun phrase. The precise nature of the constraints depends on the type of quantifier. The examples above show how this works for crisp quantifiers such as *all* or *three*. It is also possible to define constraints for gradable quantifiers such as *many* and *few*. In this case, the quantifiers are not binarily constraining the cardinality of the referent, but rather, they assign a score to every possible partitioning and work as a heuristic value in the same way as the inter- and intra-cluster variance. For example in Figure 5.4 the REFSET of the utterance "many blocks in front" refers to *cluster-2*, not because the *many* excludes *cluster-1* entirely, but because the constraint imposed by many assigns a higher score to a set of cardinality 6 than a set of cardinality 1.

For the sake of simplicity we implemented quantifiers in the same way as the spatial prototypes, using a prototypical value and a similarity function. The distance function in this case is defined as the difference between the cardinality of the REFSET and some prototypical cardinality ($\mathrm{d}(c_p, c_{\mathrm{REF}}) = |c_p - c_{\mathrm{REF}}|$). For the current experiment the average cardinality of the REFSET is around 3.5. For the purpose of the present experiment, we have chosen the prototypical values for *few* ($c_p = 1$) and *many* ($c_p = 6$) such that any cardinality above 3.5 is will be more similar to *many* and anything under 3.5 will be more similar to *few*.[1]

[1]The prototypical value of 1 may seem somewhat unnatural, we have choses this value only for the sake of this experiment. With a higher prototypical value *few* would be used much more to describe groups of objects than *many*. This asymmetry, would make it harder to interpret the results.

Figure 5.5 shows a plot of the similarity functions of *few* and *many*.

In sum, this approach regards quantifiers as constraints. They help with identifying the referent of a noun phrase. We use existing clustering methods, that model the reification of the referent as a partitioning process that is partly steered by pragmatic heuristics (e.g., inter- and intra-cluster variance) and partly by semantic heuristics (quantificational expressions).

## 5.4 Generalized Quantifiers

To measure the performance of our approach we compare it with an implementation of a common type theoretic way of dealing with quantification (Montague, 1974). These approaches commonly consider the noun (e.g., *ball*) as a predicate that together with a quantifying expression forms a (quantified) noun phrase (e.g. *all balls*). Such a noun phrase is modeled as a *generalized quantifier* (GQ) (Barwise and Cooper, 1981).

For those not familiar with generalized quantifiers, we provide a brief explanation: A noun or verb phrase denotes a property that can be represented as a function from entities to truth values, in other words, as the set of entities for which the property holds. Consequently, the interpretation of *ball* is the set of all balls $B$ in a context, and *are red* is the set of all things that are red $R$. Quantifying expressions then are understood as set relations. For instance, the sentence *all balls are red* can be modeled as $B \subseteq R$. The determined noun phrase is therefore modeled as a function from a set to truth values, in other words, a generalized quantifier. For example, the determined noun phrase *all balls* is interpreted as a function $f(Q)$ that is true iff $B \subseteq Q$. The functional role of the quantifier under this analysis is to transform the noun predicate into such a generalized quantifier. For example, the meaning of the quantifier *all* is a function $g(P, Q)$ that is true iff $P \subseteq Q$. Where $P$ is the predicate of the noun and $Q$ the predicate of the verb-phrase.

The essential restriction imposed by this approach is the fact that the noun is considered to be a predicate. In light of the previous model, this means that before applying the quantifier, we require some procedure that turns the set of scored items into a predicate (i.e., a procedure that decides for every element if it is part of the noun or not).

In accordance with this observation, we implement a model we will henceforth refer to as the *Generalized Quantifier approach*. The main difference between the Generalized Quantifier approach and our model as described earlier in this chapter lies in the operation `apply-selector` (as seen in Figure 5.2). Before applying the quantificational information, the operation establishes the set of objects that forms the extension of the noun. Just as in the previous model, the interpretation of the noun "block(s) in front of me" establishes a similarity score for every object in the context. The operation `apply-selector` determines for every object if its

score is high enough to be part of the noun. In order to make a fair comparison between the two models, we employ the exact same clustering methods for the reification of the noun as above. The main (and essential) difference with the previous model is that the reification is done *before* considering the quantificational information. This difference might seem insignificant, but it is needed to stay in line with the type theoretic approaches and, as we are about to show, has a very important impact on the performance of the model.

Generalized Quantifiers is fundamentally a theoretical proposal which does not propose any specific form of implementation. Therefore, one might be tempted to question how general our modeling is. For the concrete reification operation, other mechanisms are possible, but the main point from this section is that no matter what the particular implementation is, type theoretic approaches rely on the assumption that we can unambiguously establish the cardinality of the interpretation of the noun *without* regarding the quantificational information. The fact that there is no easy fix for this problem can be seen in the model of Fuzzy Quantifiers. This model is an fuzzy extension of GQ. Here, all set relations and predicates can be degree-based, but nonetheless, the model requires a mechanism to establish the cardinality of the fuzzy set representing the noun. This crisp intermezzo in the analysis of quantified noun phrases is needed to save the GQ representation of nouns, but makes Fuzzy Quantifiers prone to the same problem as GQ.

In sum, although type theoretic approaches do not propose a concrete operationalization, they do impose particular constraints on the way the referent is determined. The reliance on a defined cardinality for the extension of the noun is incompatible with our model as proposed in the previous section. And, as we will see in the next section, it is precisely this reliance on a defined cardinality that makes the Generalized Quantifier approach a much less suitable model for real-world application.

## 5.5 Experimental Setup and Results

Since we have operational models of the two approaches we can compare their performance in real world interactions. A population of agents play thousands of language games. Each agent is equipped with English spatial categories such as *front*, *back*, *left*, *right*, *near* and *far* and with the quantifiers *many*, *few* and the cardinals *one* to *twelve*[2] We consider two different populations of agents: in

[2]1) Syntactic processing is implemented in Fluid Construction Grammar (FCG) (Steels and De Beule, 2006). FCG maps IRL-programs to natural phrases and back given a particular lexicon and grammar. Here we implemented lexical and grammatical. Here, we equipped agents with lexical items for spatial categories (e.g., *left, back, front, right*), object classes (e.g., *block, box, robot, thing*) (Spranger, 2012) and quantifiers (e.g., *many*, *few*, *one*, *nine*). Moreover, rules for quantified adjective noun phrases, quantified noun phrases, and quantified noun phrases like "three blocks left of the box" are provided (Spranger and Steels, 2012). 2) The scenes contain

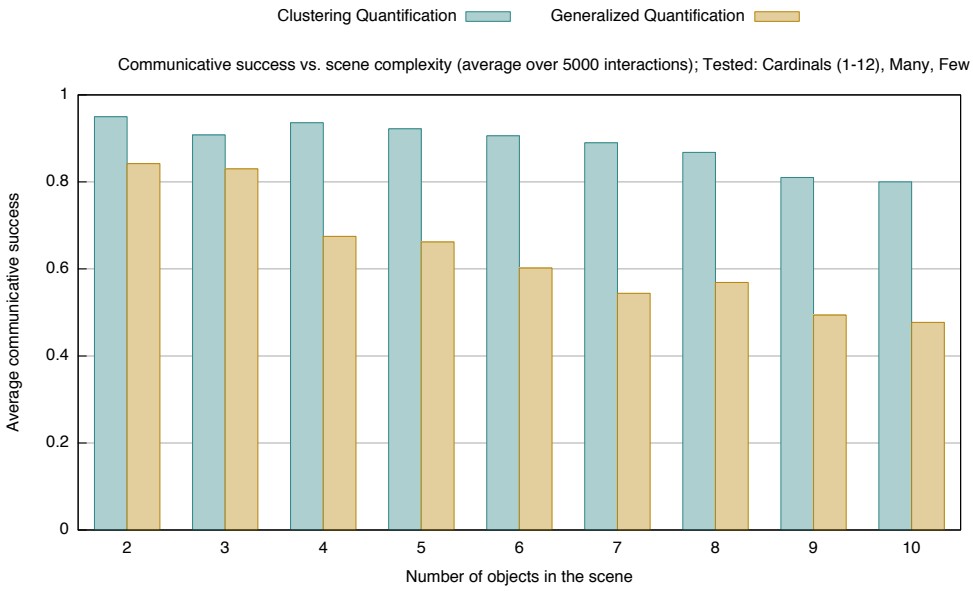



Figure 5.6: Average communicative success for 5000 interactions on different sets of spatial scenes (grouped according to number of objects in each scene).

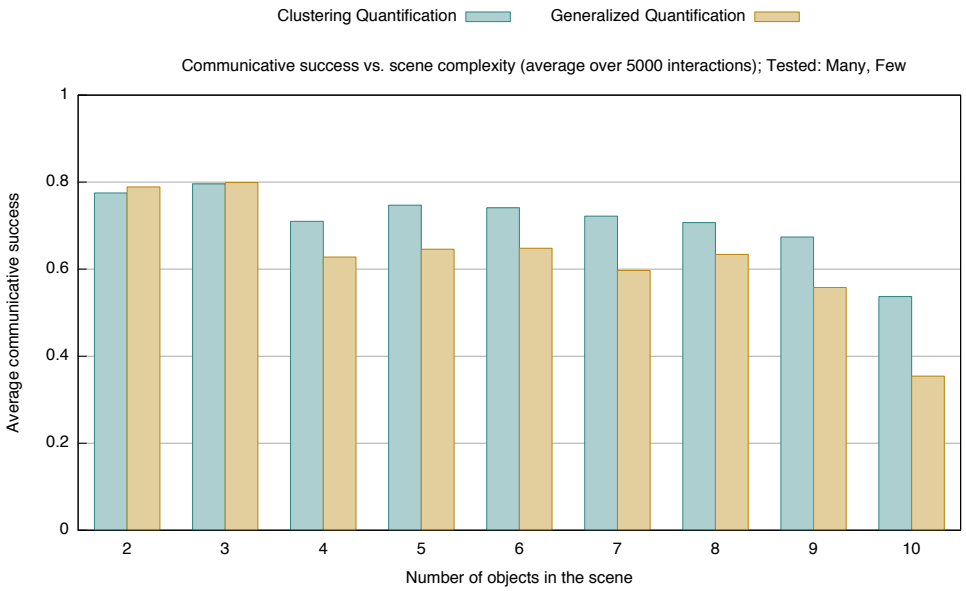



Figure 5.7: Average communicative success for populations with vague quantifiers quantifiers such as *many* and *few*.

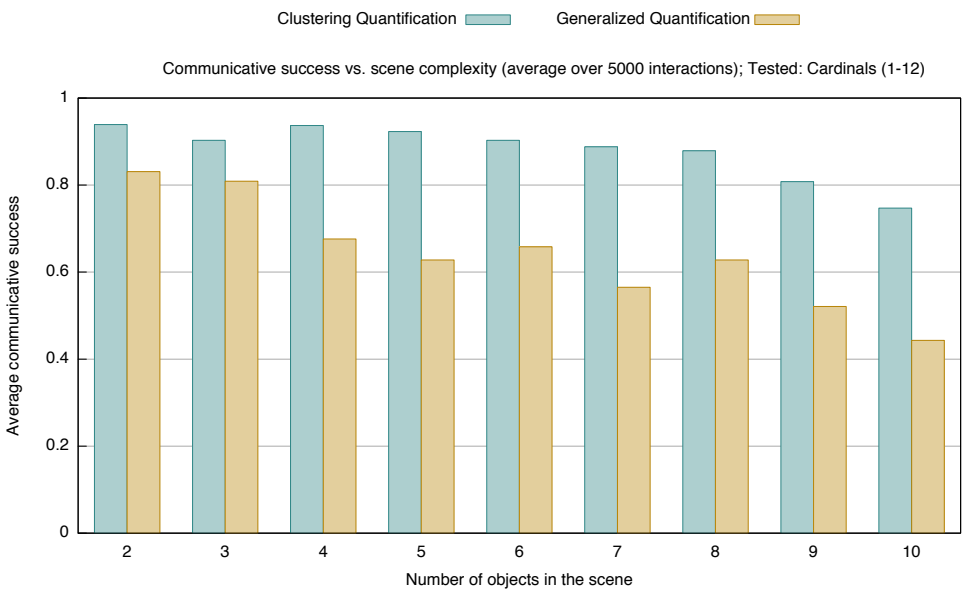



Figure 5.8: Average communicative success for cardinal only populations.

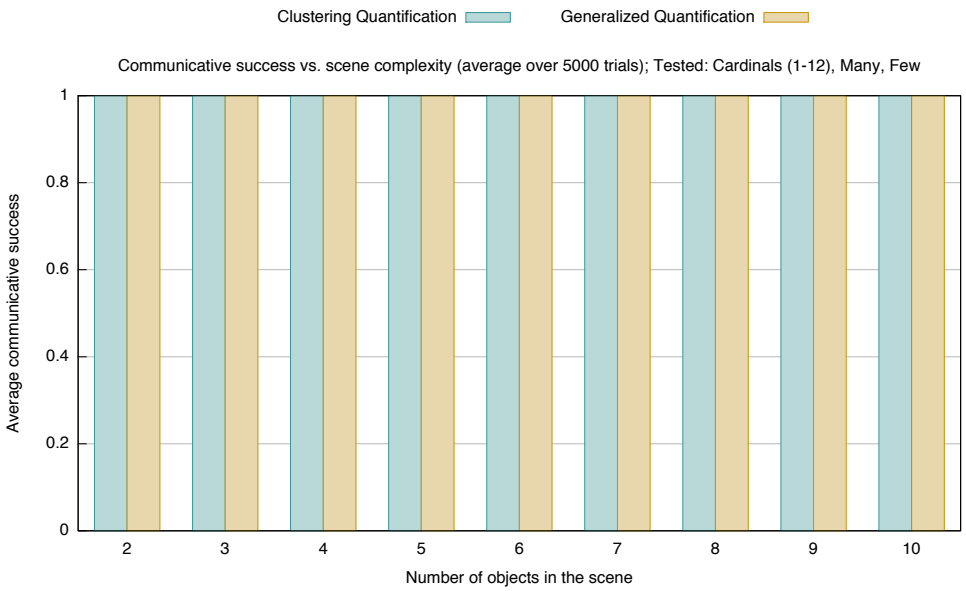



Figure 5.9: Average communicative success for populations with all quantifiers. Here both agents perceive the scene through the same camera. I.e., both agents have identical world models.

one population all agents use our model; the other population uses the generalized quantifier model. Each interaction is either successful (the hearer points

op to only ten objects. So the cardinal numbers *one* to *twelve* are enough any group of objects, even if the agent's perception is off by two objects.

to the correct set of objects) or unsuccessful (any of the steps in the language game script fails). The performance of the respective models is reflected by the average communicative success (the percentage of successful interactions) over all language games.

### Scene Complexity

Figure 5.6 shows how the two approaches perform. We test performance in different scenes. Some contain only two objects, others up to ten. The results show two important points: 1) the clustering approach performs much better than the generalized quantifier approach in all experimental conditions; and 2) increasing the complexity of the scene the difference in performance grows.

Already in the first condition, where there are only two objects in each scene, our approach reaches ca. 95% success whereas the GQ-based model reaches only ca. 85%. More strikingly though when the number of objects increases this difference grows even more. In the condition where ten objects success of GQ drops to below 50% – only every second interaction is a success. Our approach reaches 80% success even in difficult conditions.

### Cardinal vs Vague

To discern the exact performance of cardinal and vague quantifiers we tested each of them separately. Figure 5.7 shows the result for the quantifiers *few* and *many*. The results show a worse performance overall for our and the GQ model when agents can not use cardinals. Also, the difference between the two models is smaller. Only for four objects or more, the clustering approach is performing better than GQ. And, only for 9 or more objects the difference starts to be more than 10%. This contrasts with populations in which agents can only use cardinal quantifiers (see Figure 5.8). The overall average communicative success is much higher than vague-only and slightly lower than with all quantifiers. but essentially the same result is obtained. This means that cardinals are responsible for most of the communicative success when agents are given also vague quantifiers. The reason cardinals perform better than just vague quantifiers is that they communicate hard constraints. If the speaker signals he is talking about three objects this is a very clear constraint on the referent set much more so than signalling *few* or *many*.

### Perceptual Deviation

To understand why our approach performs consistently better than GQ we have to consider another condition. Figure 5.9 shows a case where the two agents interacting in a language game are perceiving the scene through the same robot body. (This manipulation is possible because software agents can access the same hardware.) In this case both approaches perform perfectly.

In embodied interactions, each agent perceives the scene through his own body. Often two agents estimate properties of the world differently. For instance, the agents in Figure 5.1 each estimate the distance and angle of objects in the scene differently. For instance, for the speaker `object-1` lies at a distance of 31.6cm and an angle of $-107$ degrees from the box. For the hearer the same object is 22.4cm away from the box at an angle of 81 degrees. We call this the problem of *perceptual deviation* (see the previous chapter or Spranger and Pauw, 2012, for more details). This problem is one of the defining characteristics of interactions in the real world. Importantly, these differences in perception affect the performance of different semantic modeling approaches.

## 5.6 Conclusion

In this chapter, we have proposed a model for the processing of quantifiers that is intended for use in real world situations. We have extended contrast and acceptability principles known from the psycholinguistics of spatial language and showed how they can be incorporated into a semantics of quantifiers that further adds cardinality constraints. We contrasted our model with a type theory based model and showed that our model 1) is more robust against the effects of perceptual deviation, and 2) scales better with respect to the complexity of scenes.

Cardinality is not the only constraint important for understanding quantifiers. Model theoretic accounts, for instance, strongly focus on the role of quantifiers for inference – a tradition that dates back as far as Aristotle's syllogisms – by considering quantifiers as a functional relation between noun and verb phrase. Our model does not deal with these aspects of quantification. It does however provide an important first step in grounding quantified noun phrases.

An important next step could be to investigate how our model of quantification holds up when used in full sentences. This would undoubtedly raise issues of reasoning and scope resolution (Kurtzman and MacDonald, 1993) and eventually its role in more complex discourse situations (Kamp, 1981b; van Eijck, 1990). In principle it is fairly straightforward to extend the current model to be used with entire sentences. IRL-networks can easily be extended to verify if a specific property (such as "is red" or "rolls") holds for (a subset of) the referent set. This way IRL-networks can be used to assign truth-values to sentences.

When it comes to inference and scope resolution, a grounded semantics approach like ours can provide important advantages. While on the one hand, grounding introduces complications, such as the problem of perceptual deviation (Spranger and Pauw, 2012), grounding does allow to resolve ambiguities by verifying different possible interpretations in the context (Spranger and Loetzsch, 2011). This reduces the need for syntactic resolution. When it comes to dealing with these concerns, the most obvious vantage point is to look at results in

Discourse Representation Theory (DRT) (Kamp, 1981b; van Eijck, 1990). All semantic entities that IRL introduces, become available as free variables throughout the entire discourse. So any referent that is being introduced can freely be used for reference later in the discourse — a treatment of referents that is quite similar to DRT.

Of course, an in depth analysis of quantification in all its complexity is well beyond the scope of this chapter. In spite of these reservations, the model does what it was designed for: It models the semantics of natural language quantifiers as expressions of quantity, grounded in real-world perception.

## Acknowledgements

We thank Masahiro Fujita, Hideki Shimomura and their team at Sony Corporation Japan for their help with the robotic setups used in the experiments. Funding for the research reported in this chapter was provided by Sony CSL Paris and the EU projects ECAgents (FP6) and ALEAR (FP7).

# Part III

# Formation Experiments

# Chapter 6

# The Emergence of Norm Dependency in Quantifiers

Human natural languages use quantifiers as ways to designate the number of objects of a set. They include numerals, such as "three", or circumscriptions, such as "a few". The latter are not only underdetermined but also context dependent. We provide a cultural-evolution explanation for the emergence of such quantifiers, focusing in particular on the role of environmental constraints on strategy choices. Through a series of situated interaction experiments, we show how a community of robotic agents can self-organize a quantification system. Different perceptions of the scene make underdetermined quantifiers useful and environments in which the distribution of objects exhibits some degree of predictability creates favorable conditions for context-dependent quantifiers. This chapter has previously been published as: Pauw, S. and Hilfery, J. (2012). The emergence of quantifiers. In Steels, L., editor, *Experiments in Cultural Language Evolution*. John Benjamins

## 6.1 Introduction

Quantifiers are ways in which the speaker can indicate the number of objects in a set. Some quantifiers are *absolute* and *precise*, such as "three". Others are absolute and *underdetermined*, such as "about three". And some quantifiers are underdetermined and *scalable* with respect to an expected number. For example, the quantifier *many* does not refer to the same amount in example 1 as it does in 2.

1. There are many students in the classroom.

2. There are many teachers in the classroom.

The intended meaning of the quantifier depends on how many students and teachers are expected in the classroom. Many factors could play a role in scaling: the size of object under consideration (Hormann, 1983; Newstead and Coventry, 2000;

Feigenson et al., 2002), their density (Coventry et al., 2005), their contour length (Clearfield and Mix, 2001), the amount of contrasting object (Coventry et al., 2010).

It has been noted both from a psychological (Sapir, 1944) and a modeling (Lappin, 2000) point of view that scalable and absolute quantifiers should be treated as different categories. Here we are interested to understand how both types of quantifiers are processed and how they could originate in a population of speakers. We focus only on one particular scaling factor, namely the expected frequency of the type of object (Moxey and Sanford, 1993a).

We follow a common approach to language evolution: Language users are assumed to acquire strategies for using, acquiring and building quantifier systems and self-organization and selection then leads the population to a shared system adapted to the ecological conditions they encounter. We have therefore operationalised two language strategies: one strategy based on absolute quantification and the other based on scalable quantification. We show that the predictability of the number of a specific type of object makes scalable quantifiers more useful and show that a selectionist dynamics at the level of strategies leads the population to adopt such a strategy when relevant.

The experiments reported here are certainly not built from scratch. They presuppose a lot of mechanisms that have been developed and tested in other related experiments (Spranger et al., 2010b; Pauw and Spranger, 2010; Steels and Loetzsch, 2009). For example we provide the agents with the mechanisms for deriving situation models through vision, for describing objects and their spatial relations and for counting the number of objects in a set. On the other hand, the experiments leave the agents free to develop their own quantifier system and use the most efficient language strategy for the environments they encounter.

In order to provide detailed insight in the dynamics of the experiments and the mechanisms involved, both language strategies are first studied in isolation. The examination of each strategy is broken down into three steps. The first step involves a reconstruction experiment. We endow the agents with a fully developed quantifier system and test its performance in a *baseline experiment*. In the second step, we introduce a set of learning operators and test their adequacy in an *acquisition experiment*. Finally, we show in a *formation experiment* how linguistic selection based on communicative success causes a quantifier system to emerge in the group through situated embodied interactions.

In a final experiment, we provide the agents with both strategies and show that, if the environment displays a consistently high degree of prototypicality, robotic agents tend towards a language strategy for scalable quantifiers. In essence, we show that environmental constraints in themselves are enough to bias the system towards scalable quantification.

We propose concrete cognitive constraints and use robots as a platform to model the use, acquisition and formation for language strategies. We do not claim that our model faithfully simulates the way humans acquire or build up

quantifier systems. We only claim that the model is functionally effective in the sense that it could be operationalized and explains the phenomena we are interested in.

## 6.2 Embodied interaction

The experiments we describe are based on communicative interactions between humanoid robots (see Steels (2012c) for specific details). Figure 6.1 shows an example scene with two robotic agents interacting in a shared environment. The setup utilized here is similar to the one used in the spatial language-game experiments as described in Spranger et al. (2010b); Pauw and Spranger (2010). Each robot perceives the world through its own onboard sensors, e.g., a camera and proprioceptive sensors. From this multimodal sensory input (Spranger, 2008), the robots build a world model, which reflects the robot's current belief about the state of the environment. Conducting experiments with actual robots is time consuming. We therefore re-use recorded data from actual robotic interactions to speed up our experiments while at the same time preserving the realism of physical robot interactions.

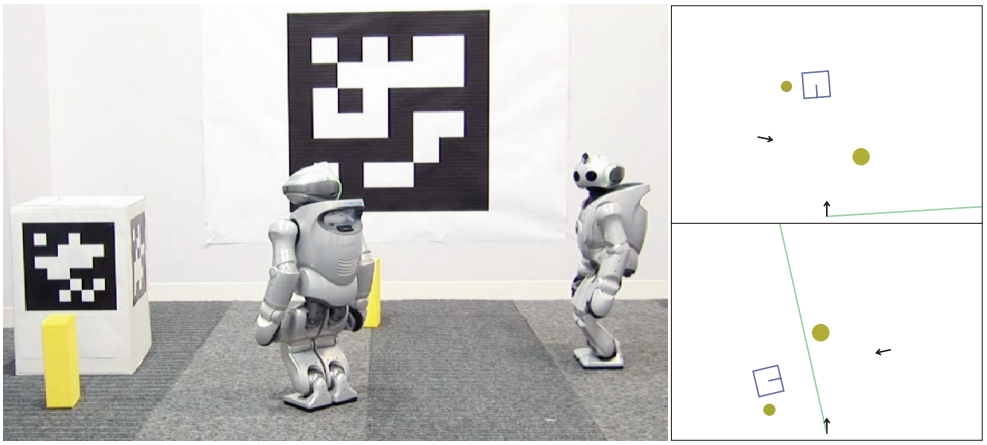
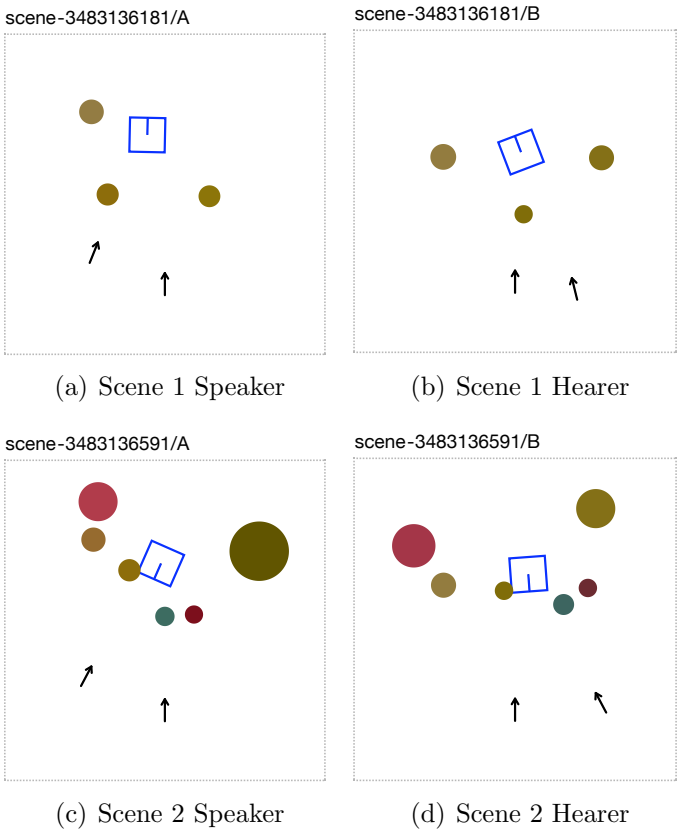


Figure 6.1: An example of a robotic interaction. Robots are placed in an office-like environment that contains different types of objects. This scene contains two yellow blocks, one box and two robots themselves. The world models of the robots are shown on the right. In the world models, the arrows represent the robots. The direction of the arrow marks the orientation of the robot. The circles represent the blocks and the blue square represents the box. The box has an inherent front (much like a car or a house – which is marked by the small blue line).

The present data set contains scenes with different types of objects: boxes, blocks, and the robots themselves. The example scene in Figure 6.1 contains one

box, two yellow blocks, and the two interacting robots. The types of objects are fairly consistent across scenes, but their number, position, and color varies. For the present experiment, it is very important to keep in mind that the number of objects, though varying, shows some level of prototypicality. In every scene, the amount of boxes ranges from 0 to 3 (averaging at 1.39), the amount of blocks can range from 0 to 9 (averaging at 4.8) and there are always two robots. It is precisely these distributional patterns that allow us to investigate scalable quantifiers.




(a) Scene 1 Speaker (b) Scene 1 Hearer

(c) Scene 2 Speaker (d) Scene 2 Hearer

Figure 6.2: Two example scenes from the points of view of the speaker and the hearer. Figure 6.2(a) shows the internal representation of Scene 1 for the hearer. Figure 6.2(b) shows how the hearer perceives the same scene. Figures 6.2(c) and 6.2(d) display the internal representations for scene 2. The arrows represent the robots (with their respective orientation). The circles represent the blocks (with their perceived size and color). The blue square represents the box.

The agents use a conceptualization system known as IRL (or "Incremental Recruitment Language"). IRL is a procedural semantics in the same spirit as Winograd's SHRDLU framework (Winograd, 1980). Conceptualization in such a system is a planning process that incrementally puts together cognitive operations in order to fulfill a specific communicative goal (the interested reader is referred

to Steels and Bleys (2005); Van Den Broeck (2008); Spranger et al. (2010a) and Chapter 3 of this thesis for the specifics of this approach.

In the present experimental design, we use a description game similar to the one described in van Trijp (2008): the speaker and its interlocutor are not presented with a single scene but with a number of scenes. One of the scenes is what we will call the **focus scene**. The speaker attempts to find an utterance that establishes a referent in the focus scene but not in the other scenes. Consider, for example, Figure 6.2. The utterance "some blocks in front of me" has a referent in the focus scene (Scene 1) but not in the other scene. The interlocutor in turn has to interpret the utterance and find the scene which contains a referent for that utterance. We build upon an existing system for spatial language (Spranger et al., 2010b). This framework employs a prototype system (Rosch et al., 2004; Lakoff, 1987) that allows the agents to calculate the representivity of both the objects that populate their surroundings as well as the spatial relations that occur between these objects. This allows the robots to distill the correct referent of "the block left of the box" or "the box in front of me."

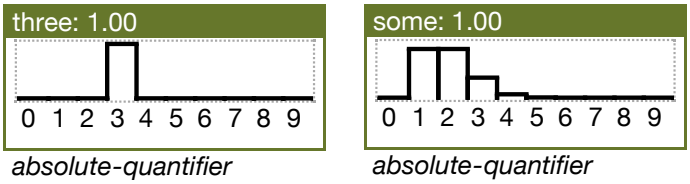
## 6.3 Experiment 1: Absolute quantification

The model for spatial language establishes how well an utterance such as "block in front of me" describes an object in a given context. In complex environments, however, there can be more than one exemplar that answers to a given description. In such cases, the agents need to be able to deal with sets (and subsets) of particular objects. Pauw and Spranger (2010) show how the quantificational aspects of an utterance provide information for this selection process. Consider the utterance "the three blocks in front of me." The semantic operations that are associated with *the three* signals that the speaker has a unique referent set in mind of cardinality 3. Using this information the robots simply pick the three elements from the context that best fit the description "blocks in front of me." However, for fuzzy quantifiers such as *about three*, the matter is somewhat less straightforward.

The model we use is based on Zadeh's fuzzy-quantification mode (Zadeh (1983) - see Glöckner (2006) for a recent overview). This model relies on the observation that most natural-language-type quantifiers (e.g., *some* or *many*) cannot be interpreted in terms of absolute truth. In other words, the sentence "Some blocks are red" is true to a certain degree. If there are three blocks in the context that are red, the utterance is 'more true' than if we utter the same sentence in a context where there are five red blocks. The degree of truth is indicated by a score between 1 and 0, where 1 is completely, undoubtedly true and 0 means that the utterance is entirely false. The meaning of *some* can be established by answering the question: How true is the sentence "there are some blocks" (with regard to a specific scene)? If there are three blocks in the scene,

we assign a score of 1. If there are four, a score of 0.8. If we do so for every cardinality, we can establish a distribution of scores that represent the meaning of the quantifier *some* (see Figure 6.3). Given such a scenario, the quantifier *some* might be used to allude to a referent of cardinality 2 more than it would to a referent of cardinality 4. In this sense, the utterance "Some blocks are red" is more readily true in a context where there are two red blocks than in a context with four red blocks.

Crisp quantifiers can also be represented as distributions. However, in such cases, the values of the bins are either 1 or 0 (and nothing in between). Hence, the quantifier *three* assigns a score of 1 to the cardinality 3 and a score of 0 to all other cardinalities. Under such an analysis, crisp quantifiers are merely a special case of fuzzy quantifiers.



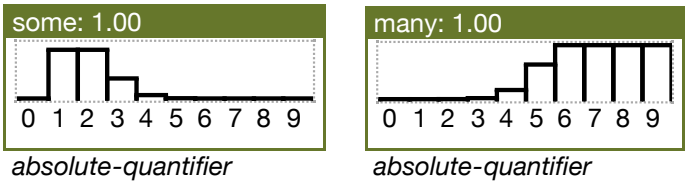

(a) Three (b) Some

Figure 6.3: Representations for the expressions some and three. The height of the bars in the distribution reflect the scores for the associated cardinality (marked below the bar). The quantifier three (a) assigns a score of 1 to cardinal 3 and a score of 0 to all other cardinalities. The quantifier some (b) shows a more gradual curve.

### 6.3.1 Baseline experiment

Before understanding how a system of fuzzy quantifiers can be self-organized by a population of agents, we need to look at the behavior of a fully developed quantificational system. We scaffold the agents with two fuzzy quantifiers representing *some* and *many*. Figure 6.4 shows the chosen interpretation of these quantifiers. For now, the agents do not invent new language items or learn anything from other agents.

We test the performance of the quantifiers by letting two robotic agents play a series of language games as described above. The agents are confronted with two scenes, one of which is the focus scene. With every interaction in the language game, one agent takes the role as a speaker, and the other, that of the hearer. The speaker thus tries to find an utterance that has a referent in the focus scene but not in the alternative scene. After interpreting this utterance, the hearer points at the scene that it thinks the speaker is alluding to. The game is a success if the




(a) Some (b) Many

Figure 6.4: This figure shows the baseline interpretation of quantifiers some and many if their scalar effects are disregarded. According to this "absolute" definition, six objects are considered many objects, two objects are considered some objects, and four is a dubious case.

hearer points correctly. For example, the speaker could use the utterance "Some blocks in front of me", to establish Scene 1 in Figure 6.2 as the focus scene.

Figure 6.5 shows the results of the communication game. Every bar describes the average communicative success over 500 interactions. There are three different conditions. In the first, we let agents randomly pick from a set of recorded scenes. Since the amount of objects in every scene is limited, there is a fair chance that quantity alone cannot act as a differentiating factor between scenes. Given such an environmental limitation the communication can never reach a 100% success rate. In fact, we observe that for randomly picked scenes, the speaker manages to correctly describe a scene in about 80% of the cases.

The second and third conditions show what happens if we cherry pick scenes such that the objects of one specific type are guaranteed to differ in both scenes. For the second condition we ensure that the scenes never have the same amount of blocks. And for the third, condition we ensure that the amount of boxes always differ. We can see that, with respect to the baseline, the communicative success improves for the second condition (reaching about 95%). The third condition, however, does not yield any clear improvement. Overall the scenes typically contain between 1 and 3 boxes (averaging at 1.2 boxes per scene). This is always best described as *some* boxes according to the absolute interpretation of *some*. Therefore, with absolute quantifiers, the number of boxes cannot be used as a differentiating factor. The next chapter shows that this can be solved using scaled quantifiers.

The results of this experiment are threefold. First of all, we show how communicative success is influenced by manipulating the statistical properties of the scenes. Secondly, we will use this data in the next section to show that scaled determiners perform much better in some cases. Thirdly, it establishes a baseline for the rest of the experiments (80%). This is important because it will allow us to test the performance of the learning operators in the next experiment.

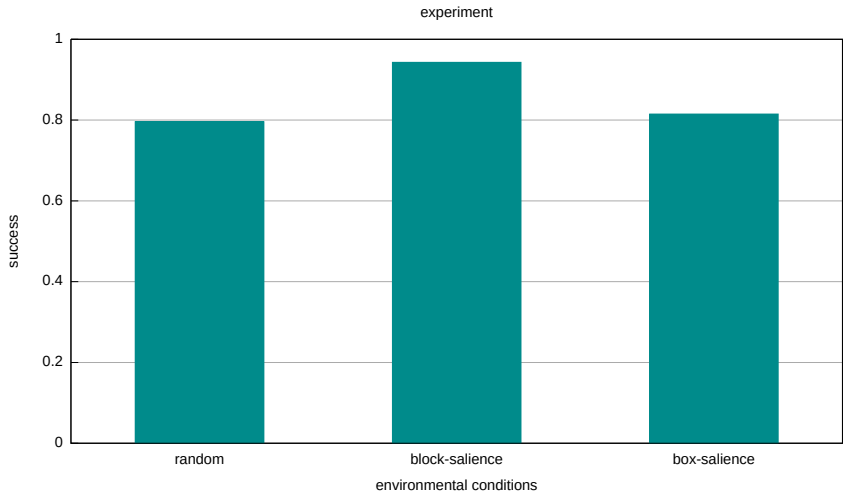



Figure 6.5: Baseline communicative success for absolute quantifiers. This graph shows the average communicative success after 500 interactions for three different conditions. The first condition shows the average communicative success when the agents are presented with two randomly selected scenes for every interaction. Under the second condition, the scenes are cherry picked such that they do not contain the same amount of blocks in every interaction. For the third conditions, the amount of boxes differs.

### 6.3.2 Acquisition experiment

We now turn to the acquisition of quantifying expressions. How can one agent (learner) learn the absolute quantifiers of another (tutor) agent. To study this, we scaffold only a tutor agent with quantifiers. The learner agent does not know any quantifiers yet, but is provided with learning operators that enable him to acquire their meanings over a series of interactions. This experiment follows the same general approach as all the other acquisition experiments in the field. There are two learning operators: an acquisition operator, that is used when the learner hears a specific linguistic construct for the first time; and an alignment operator that allows the agent to gradually shape the meaning of a linguistic construct.

### 6.3.3 Acquisition operator

The acquisition operator is only used when an agent has never heard a specific quantifying expression before. For now, we presuppose that the agent knows that the unknown lexical item is an absolute quantifier. The agent has to find a quantifier that would isolate a referent in the focus scene, but not in any of the other scenes. Such a quantifier is fairly straightforward to establish. Take for example, Figure 6.2. The utterance "three blocks in front of me" has a referent in the focus scene but not in the alternative scene. Thus, if the utterance was "some blocks in front of me" and the hearer knows which of the scenes is the focus scene, it can establish that 3 is an appropriate cardinal for the quantifier *some*. The acquisition operator simply finds the cardinality that would best discriminate the focus scene and creates a new quantifier with the value 1 for this cardinality and a value 0 for all other cardinalities. Figure 6.6 shows the result of the acquisition operator for this specific example. Of course this initial distribution only very poorly reflects the actual meaning of the word *some*. The alignment operator will gradually improve the meaning over many interactions.

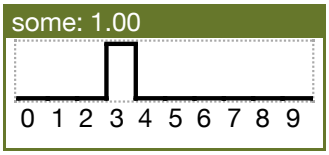



Figure 6.6: The result of the acquisition operator. This is the meaning the learner agent acquired for the word *some* with the utterance "some blocks in front of me" in the context shown in Figure 6.2. While the acquisition operator does not find a perfect interpretation right away (i.e., some does not mean 'precisely three'), the alignment operator gradually shapes the representation to accord with that of other agents.

### 6.3.4 Unbiased alignment operator

The alignment operator is triggered after every unsuccessful interaction. This presupposes that the failure of the interaction was due to a poor alignment of the quantifier used in the communicative event. As a first step, a new quantifier is computed in precisely the same way as the acquisition operator does, disregarding any previously established meaning. This new quantifier is then merged with the existing meaning. The merging is done by looping over every bin in the original meaning and the corresponding bin in the new quantifier. A new value for the bin is computed as a mixture by the following function:

$$Q(i)_{n+1} = (1 - r_a)Q(i)_n + r_a Q(i)_c \tag{6.1}$$

where $Q(i)$ denotes the value of bin $i$ for quantifier $Q$. Here, $Q_{n+1}$ is the updated quantifier, $Q_n$ the previously learned quantifier, and $Q_c$ the new quantifier. The learning rate, $r_a$, determines at what rate the new observations influence the old ones. If $r_a = 0$ then the alignment operator will have no effect whatsoever. If $r_a = 1$ the old quantifier will be completely replaced by the new observation. Figure 6.7 shows an example result of such an alignment iteration.

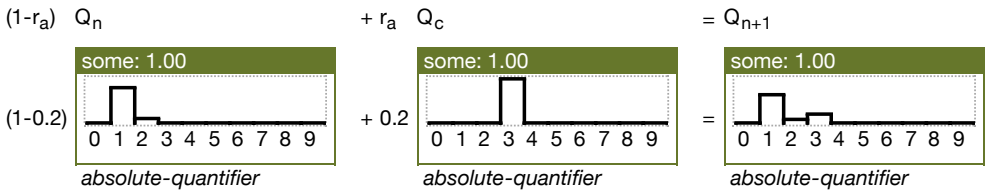



Figure 6.7: The result of the alignment operator. The learner agent finds evidence that "some somethings" can *also* mean "3 somethings" (apart from what it learned already before about the quantifier *some*). Quantifier $Q_c$ represents the newly acquired information, $Q_n$ the current representation and $Q_{n+1}$ the result of the alignment process.

Figure 6.8 shows the results of an acquisition experiment. The (increasing) communicative success is plotted over 800 interactions. The graph also shows the baseline success established in the previous experiment (i.e., the maximum communicative success that the agents can reach) and the variance between concepts. The measure of variance is based on statistical variance. For two quantifiers $Q_1$ and $Q_2$ the variance $Var(Q_1, Q_2) = \sum_{i=1}^{n}(Q_1(i) - Q_2(i))^2/n$ — where $n$ is the range of the quantifier. As can be seen in the graph, communicative success does not reach the baseline and the variance stays at 1.7. In other words, the learner does not properly align its concepts with the tutor. The main problem is that the learner finds too little consistent evidence to learn the tutor's concepts to a "satisficing" level.

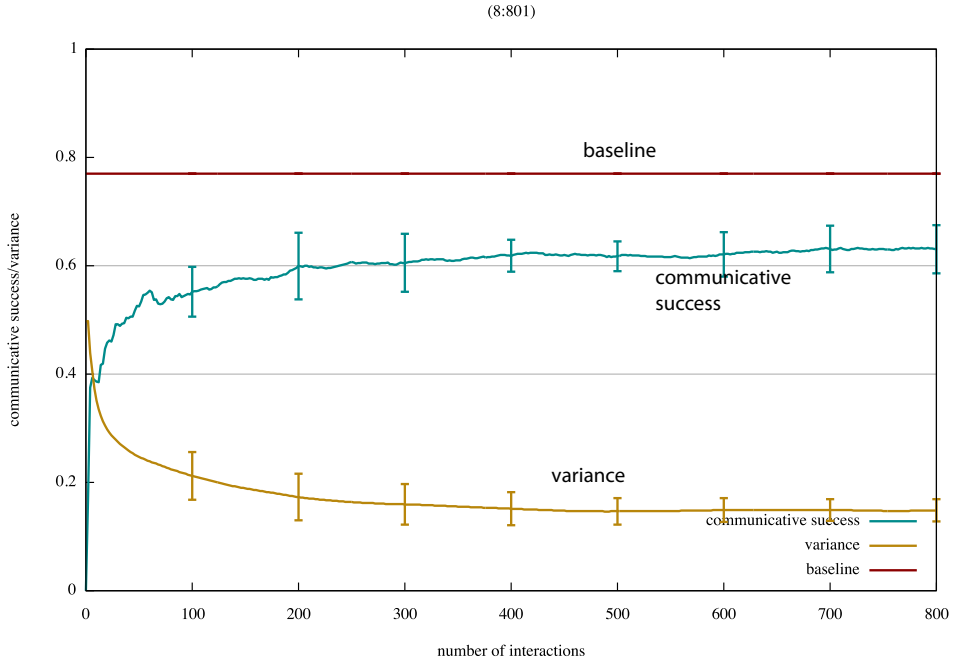



Figure 6.8: Learning absolute quantifiers without any bias. The graph shows the average result of 6 trials of 800 interactions for a community of 5 agents. The communicative success increases but does not reach the previously established baseline of 77% communicative success. The variance stagnates at 1.7, which is not enough for optimal communication.

### 6.3.5 Aligment with convexity assumption

The number of possible quantifiers that can be expressed by distributions as discussed above is in principle infinite since the bins can contain any value between 1 and 0. In practice this number is bounded because, in order to achieve communicative success, quantifiers do not have to be identical but just similar enough. However, the number of possible (distinct enough) quantifiers increases exponentially with the number of objects in the environment. It is therefore not surprising that the quantifiers have to be constrained in order to be learnable (see also van Rooij (2006)).

To constrain the possible quantifiers to those that might only naturally occur in human language, we make use of the *convexity* principle (Gärdenfors, 2004). A convex property is a domain-independent property that does not have any "holes." For example, the concept *left* is convex, because if two points belong to the category *left* than any point in between also belongs to this category. The color *red* is another example of a convex concept: if two different hues are classified as red than any hue in the spectrum inbetween is also classified as red. Gardenförs shows that there is compelling evidence that natural properties are convex.[1]

In the case of quantifiers, the meaning of *an even number of* is not convex. The number 2 is even, the number 4 is even, but the number 3 is not. Thus, the distribution would show a hole at number 3 (and at 5, 7, etc. . . ). The quantifier *some*, on the other hand is convex. If in a specific context, *some blocks* can refer to both five blocks and seven blocks, then per force, *some blocks* must refer to six blocks as well.

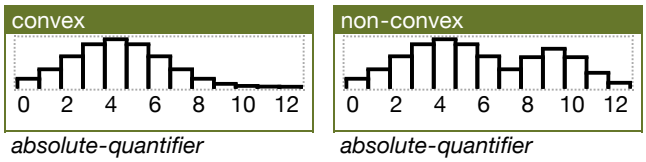



Figure 6.9: Convex quantifiers have only one peak. The distribution on the left represents a convex quantifier. The one on the right is nonconvex.

In the case of fuzzy quantifiers, such a hole appears as a valley between two

[1]We of course recognize that convexity is not the only way to constrain quantifiers. For example many applications of prototype theory assume a prototypical value and a score that depends on the distance to this prototype (normally according to a bell-curve). Another way to constrain the quantifiers would be by assuming some (skewed) normal distribution of the scores, or assume that they can be described by Zadeh's S-function (Zadeh, 1983). In any event, all of these assumptions ultimately entail convexity at the price of making more assumptions about the specific shape of the curve. The reason to employ convexity is that it is a minimal cognitively valid assumption.

peaks. Thus, a convex fuzzy quantifier has only one peak. Figure 6.9 shows the difference between a convex and a nonconvex quantifier. Gardenförs initial formulation of convexity is intended for crisp category membership. For the fuzzy case, we use Zadeh's definition of convexity (Zadeh (1965)) for fuzzy sets. Applied directly to the fuzzy quantifiers we get that:

$$\text{Quantifier } Q \text{ is convex } \textit{iff}\text{: If } x \leq y \leq z \text{ then } Q(y) \geq \min[Q(x), Q(z)] \quad (6.2)$$

where $Q(x)$ is the value of bin $x$ for quantifier $Q$. For example, if the quantifier *some* assigns as score of 0.8 to cardinality 3 and a score of 0.4 to cardinality 5, then the score of cardinality 4 has to be higher then the lowest of the two scores 0.8 and 0.4 (hence, higher than 0.4). One can verify easily that this is a minimal definition to restrict fuzzy quantifiers to only one peak.

To employ the convexity assumption in the experiment, we introduce a convexity operator that keeps all the quantifiers convex. The convexity operator is applied directly after the previously introduced alignment operator. It essentially functions as a fitting operator that takes the result of the alignment operator ($Q_{aligned}$) and finds a quantifier ($Q_{convex}$) that is convex and fits the updated quantifier as well as possible.[2] Figure 6.10 shows an example application of the convexity operator.

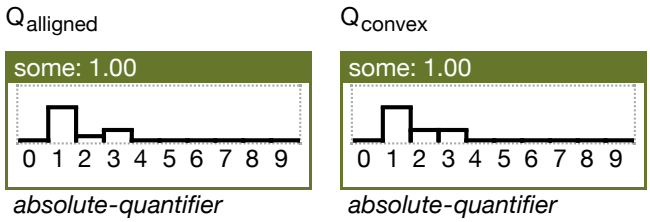



Figure 6.10: The result of the convexity operator. After the alignment operation produced a new quantifier ($Q_{alligned}$), the convexity operator finds a quantifier $Q_{convex}$ that is convex and resembles the original aligned quantifier (using a least squares approximation)

Figure 6.11 shows a graph for the same acquisition experiment as described above, but now with use of the convexity operator. The results show that the communicative success reaches the baseline.

In sum, if we make no assumptions about the representation of natural quantifiers, the dimensionality of the learning problem is too high for the amount and dependability of the evidence that the agent gather. The convexity assumption reduces the degrees of freedom enough to allow for almost perfect alignment.

[2]More precisely, $Q_{convex}$ is a convex quantifier such that the squared distance between $Q_{convex}$ and $Q_{aligned}$ is minimized (i.e., the value $sum_{i=0}^{n}(Q_{convex}(i) - Q_{aligned}(i))^2$ — where $n$ is the range of the quantifier). Note that there can be more then one solution. The precise solution that is chosen, does not appear to have any effects on the end result.

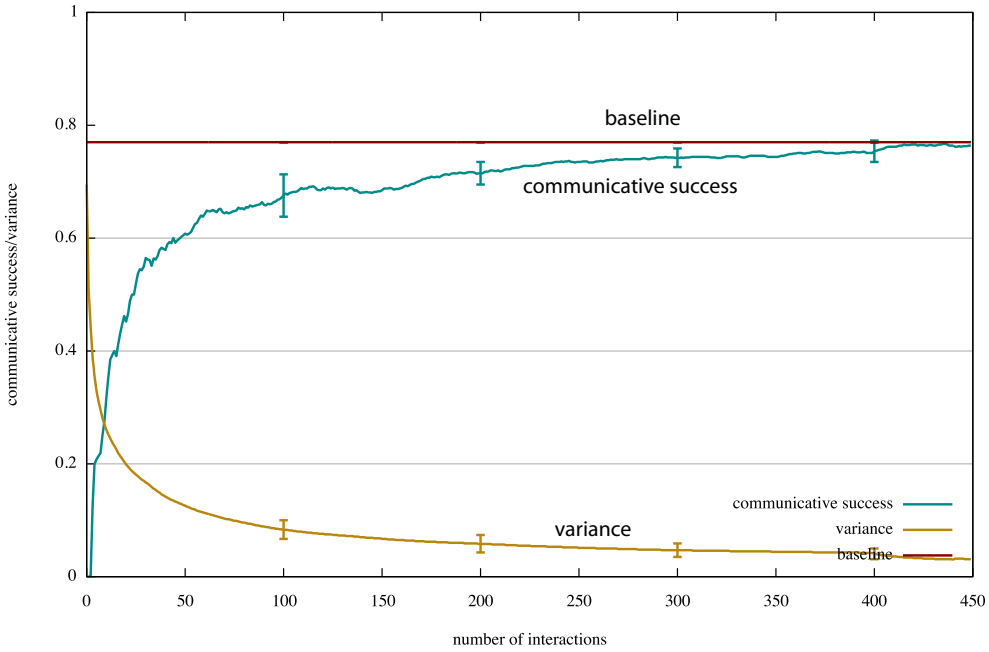



Figure 6.11: Learning absolute quantifiers with convexity assumption. The graph shows 6 trials of 450 interactions for a community of 5 agents.. The communicative success increases and reaches the previously established baseline of 77% communicative success. The graph also shows how the representations of the learner and of the tutor become more similar over time (variance).

### 6.3.6 Formation experiment

The previous section establishes the required operators that enable agents to align their linguistic concepts of absolute fuzzy quantifiers. Now we can turn to the question of how such concepts come to exist. As outlined in Steels (2012c), linguistic innovation is a mirror image of learning. To achieve the communicative goal (establishing a focus scene), the speaker goes through a planning process. It is possible with the semantic concepts and procedures presented here that the planning process cannot find an appropriate description for the focus scene. In such a case, the agent invokes an *invention operator*.

The invention operator consists of two parts: When the speaker realizes that the current set of quantifiers is not expressive enough, it can choose to either invent a whole new quantifier or to adapt existing ones. In the first case, the agent invents a new quantifier using the same acquisition operator as described in the previous section. Similarly if the speaker chooses to adapt its existing quantifiers, it simply employs the alignment operator. After the introduction of a new quantifier, the learning operators will ensure that the invented quantifiers will become shared by the entire population of agents when they are sufficiently successful.

We scaffold the agents also with a memory constraint that blocks the possibility to develop an unbounded set of quantifiers. There are three reasons for this: 1) Commensurability: It is important to keep the results of the experiments throughout the chapter comparable; 2) Tractibility: The computational complexity increases significantly with the number of available semantic concepts. To be able to run the experiments in reasonable time, we limit the number of quantifiers; And 3), plausibility: Allowing the agents to develop very specific concepts for every possible situation would make them very successful in the long run, but would have no correlate in human language.

For every quantifier that the agents introduce a score is kept that reflects how often the category is successfully applied. If this score drops below a specific threshold, the quantifier is removed from the agent's ontology, making room for other new candidates. As discussed above, the agents are biased to only develop two quantifiers. This is achieved by limiting the sum of the scores of all quantifiers together to two.

Figure 6.12 shows the results of our experiment. Note that the agents manage to develop a stable quantifier system on the basis of the selectionistic processes: invention of items, their adaptation and, if they are not successful, their elimination. The graph clearly shows that communicative success converges to the baseline success. This is the maximum they can achieve with only two quantifiers, as the quantifiers in the baseline experiment were already chosen to be optimal.

Observe also that the graph shows an initial overshoot of newly invented quantifiers. The robots try out all kinds of new quantifiers until they hit upon a set that proves successful over time. When a set of successful quantifiers is found,

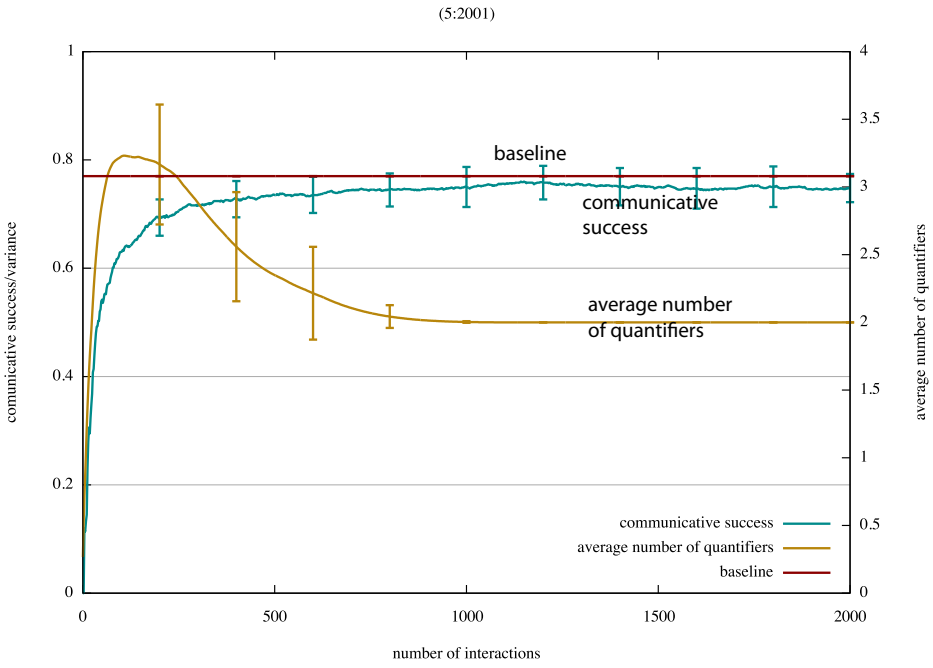



Figure 6.12: Self-orgaization of absolute quantifiers with convexity assumption. The graph show the average result of 5 batches of 2000 interactions for a community of 5 agents. The graphs shows an increase of communicative success that converges on the baseline success after 1000 interactions. We see an initial overshoot of lexicon size. After about 300 interactions, the amount of quantifying expressions starts to decrease until after 1000 interactions the agents converge on a set of two optimal quantifiers.

the nonsuccessful will automatically phase out.

Figure 6.13 demonstrates that the agents converge on an optimal set of quantifiers. This is a very important point. The graph shows a quantifier (*sosa-3*, meaning 'an average amount of') that is individually successful, but eventually does not survive, because it does not perform well in combination with other quantifiers. Eventually the agents converge on a set of two quantifiers (*casa-81* and *kosa-4*, meaning respectively 'some' and 'many') that are more successful in combination with each other.

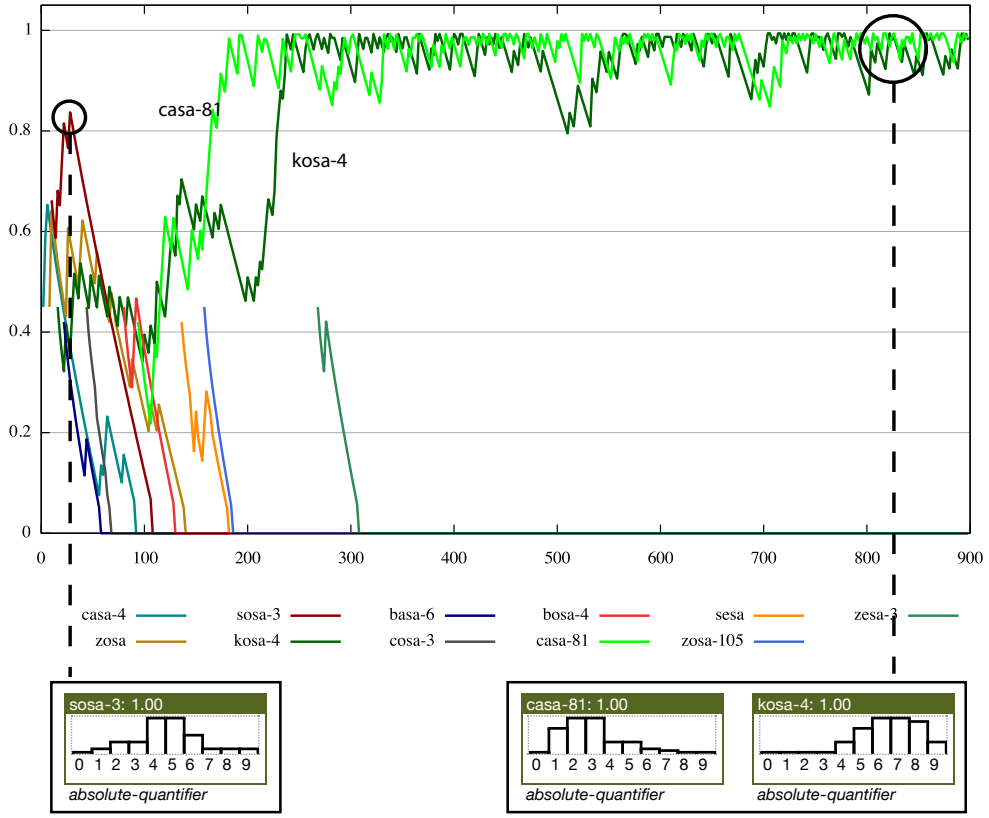



Figure 6.13: This graph shows the development of the confidence scores of every quantifier over a series of 900 interactions in a single agent. Note that the quantifier sosa-3 seems to be successful in the beginning, but does not work well in combination with other quantifiers. After 200 interactions, two successful quantifiers start to emerge (casa-81 or kosa-4) replacing sosa-3.

In sum, this experiment shows that, the agents can self-organize a quantifier system that reaches the baseline communicative success as established above. Since the quantifiers in the baseline experiment are chosen to be optimal for the present data, this suggests that our learning and invention operators — which only take individual items into account — are sufficient to allow the agents to

develop a globally optimal set of quantifiers. Hence, though the operators are defined locally (i.e., they never consider the system as a whole), the overall system converges on a globally optimal solution.

## 6.4 Experiment 2: Scalable quantification

The previous experiment shows the dynamics of the absolute quantifier strategy. Now we will look at the dynamics of scalable quantifiers. Recall that scalable quantifiers are those which depend on a specific, context-dependent norm. For instance, what counts as *many* in "many birds in a flock" is not likely to be the same amount as *many* in "many lions in a pride". For the following experiment, we assume, for the sake of simplicity, that the norm of the quantifiers *some* and *many* is solely based on the type of object under discussion.

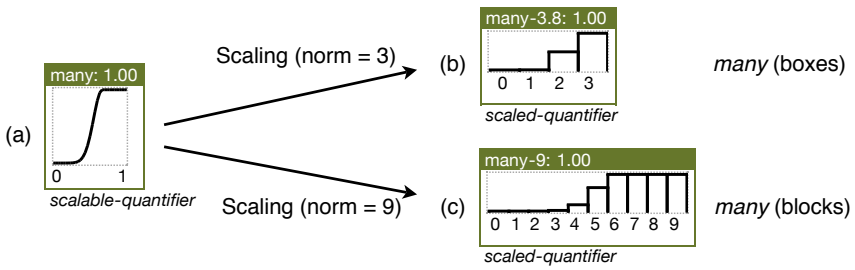



Figure 6.14: The scalable quantifier (a) is a function that assigns scores between 0 and 1 over the range from 0 to 1. The context-dependent (i.e., scaled) interpretation is calculated by a scaling operation, which takes a representative number of samples from the original, scalable quantifier. The number of samples depends on the norm inherent in the context. Figures (b) and (c) show the results of the scaling operations for the norms 3 and 9, respectively.

Just as with absolute quantifiers, we represent scalable quantifiers as distributions. However, the robots use a special scaling operation that adjusts the quantifier to a specific norm. In this experiment, the norm is provided by the object class. Every quantifier has a base representation that can be seen as a function that assigns scores between 0 and 1 over the range 0 to 1. This can be seen in Figure 6.14. The scaling operation samples values from the scalable quantifier at an interval depending on the norm. For some norm $n$, the scaling of the base quantifier $Q_b$ to the scaled quantifier $Q_n$ is given by the following equation:

$$\begin{aligned} Q_n(x) &= Q_b(x/n) \quad && \text{for } x \leq n \\ Q_n(x) &= Q_b(1) \quad && \text{otherwise} \end{aligned} \tag{6.3}$$

Above, a scalable quantifier is characterized as a function. In practice, however, they are approximated by a discrete distribution, where the bins provide sample values in the range of 0 and 1. The intermediate values are determined by linear interpolation. Thus, the number of bins does not determine the range of the quantifier, but its resolution. Representing the scalable quantifiers in the same way as absolute quantifiers gives a clear advantage: We can use the same learning operators for both types of quantifiers. This is not only an advantage for simplicity's sake, but also makes it easier to compare the dynamics of acquisition and evolution for both strategies.

### 6.4.1 Baseline experiment

How does the above relate to the absolute version of fuzzy quantifiers? In order to compare the two models, we repeat the same series of experiments for scaled quantifiers: First we establish the baseline communicative success, followed by an acquisition and an evolution experiment.

For the baseline experiment, we provide the agents with two scalable determiners representing *some* and *many* and a norm for every object class. The values are selected to be optimal. In short, this means that no other set of quantifiers could achieve a higher communicative success for the given data (see Figure 6.15).

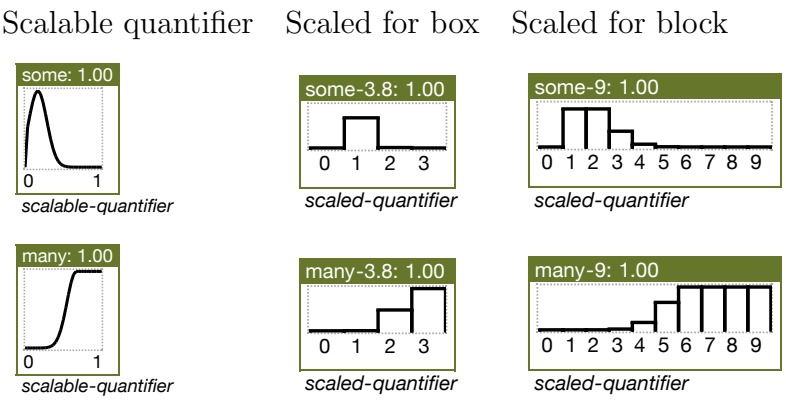



Figure 6.15: This figure shows how *some* and *many* scale to the object classes *box* and *block*. For example, three boxes are considered to be many boxes, but three blocks not.

The robotic agents play the same language game as in previous baseline experiment (see section 6.3.1). The speaker has to draw the hearer's attention to a focus scene by finding an utterance that only has a referent in that scene. The agents do not try to invent new language constructions or learn the language of the other agents. The only objective of this experiment is to establish a baseline for communicative success.

Figure 6.16 shows the results of the communication game. As a comparison, the previous results of the baseline experiment for absolute quantifiers are shown in the same figure. Every bar describes the average communicative success over 500 interactions. There are three different conditions: In the first, we randomly pick scenes from our data set; for the second condition, we ensure that the scenes do not have the same number of blocks; and for the third condition, we ensure that the number of boxes differ.

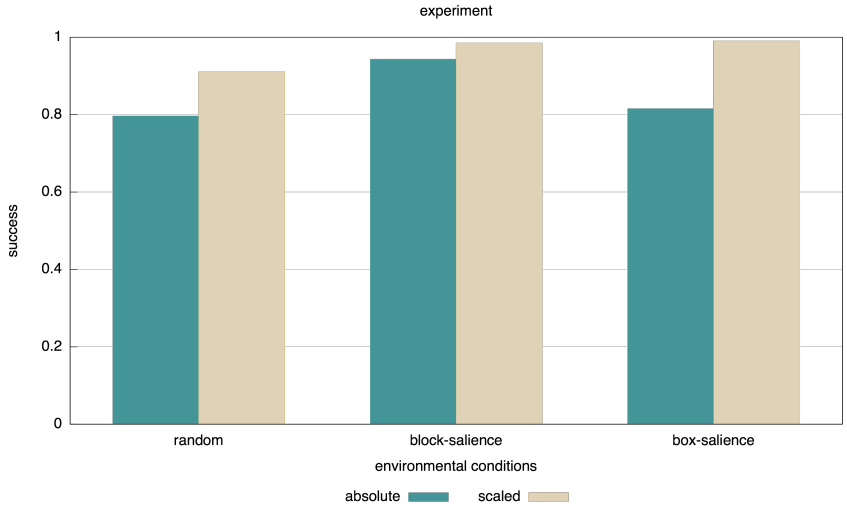



Figure 6.16: Baseline communicative success for absolute quantifiers. This graph show the average communicative success after 500 interactions for three different conditions. The first condition shows the average communicative success when the agents are presented with two randomly selected scenes at every interaction. For the second condition, the scenes are cherry picked such that they do not contain the same amount of blocks. For the third the amount of boxes differ.

We can see that the scaled quantifiers perform better for all three conditions (90% vs 85% for the random scene condition, 99% vs 95% for the block condition and almost 100% vs 81% for the box condition). We see a general increase in communicative success and especially so for the box condition.

Scaled quantifiers can be adapted to the expected amount of any type of object. Since, in our data there are never more than three boxes, the scaled quantifiers will classify two boxes as *many*. The upshot of this is that the scalable quantifiers are equally applicable to the number of boxes as to the number of blocks. Scalable quantifiers therefore seem to be more versatile than absolute quantifiers in the sense that they can be used in many more types of situations. In this experiment we see how this leads to higher communicative success.

### 6.4.2 Acquisition experiment

The learning operators for scalable quantifiers are more complex then those of absolute quantifiers. The meaning of the scalable quantifiers depends on a specific norm. Recall that, in our experiment, the norm is a prototypical scaling factor for a particular object type. The acquisition and alignment operators have to take this norm into consideration. This is realized by what we call a *normalization* operator. The normalization operator does the inverse of the scaling operator: given a norm and an absolute quantifier, normalization computes the associated scalable quantifier.

As shown above, the acquisition operator for absolute quantifiers computes an absolute quantifier for the context at hand. For scalable quantifiers, however, the acquisition operator uses the normalization operator to translate the resultant absolute quantifier into a scalable quantifier. Figure 6.17 exemplifies the acquisition of the quantifier *some*.

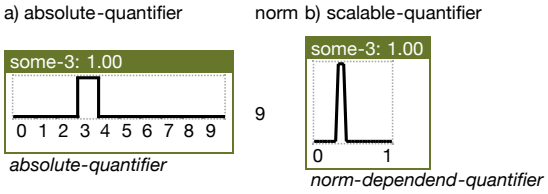



Figure 6.17: Extension of the acquisition operator for scalable quantifiers. The base quantifier is the result of applying the normalization operator to the newly acquired absolute interpretation of some. The norm in this figure is 9, which is the norm we established for the block object class.

As with absolute quantification, with scalable quantifiers, the alignment operator builds upon the acquisition operator. If the communicative interaction fails, the acquisition operator is used to create an alternative hypothesis for the meaning of the quantifier. This alternative is subsequently merged with the initial meaning of the quantifier in question. The merging procedure is identical to the one used for absolute quantifiers. This is possible because we represent scalable quantifiers in the same way as absolute quantifiers. Figure 6.18 shows an example of this procedure.

Figure 6.19 shows the results of an acquisition experiment using these procedures. As predicted by the baseline experiment, the agents achieve a higher communicative success with scaled quantifiers than with absolute quantifiers. The point here, then, is that our agents achieve near-perfect alignment using the exact same procedures as with absolute quantifiers. This guarantees commensurability between the proposed experiments.

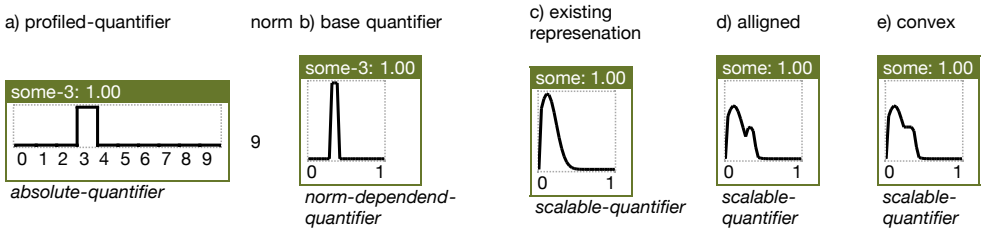



Figure 6.18: Extension of the alignment operator for scalable quantifiers. This figure shows all the steps of the alignment operator. The agent finds the most appropriate quantifier (a). This quantifiers is normalized with respect to a given norm (b). The existing representation of the quantifier (c) is merged with the newly acquired base quantifier (d). The convexity operator is applied (e). As in Figure 6.17, the norm in this figure is 9, which is the norm we established for the block object class.

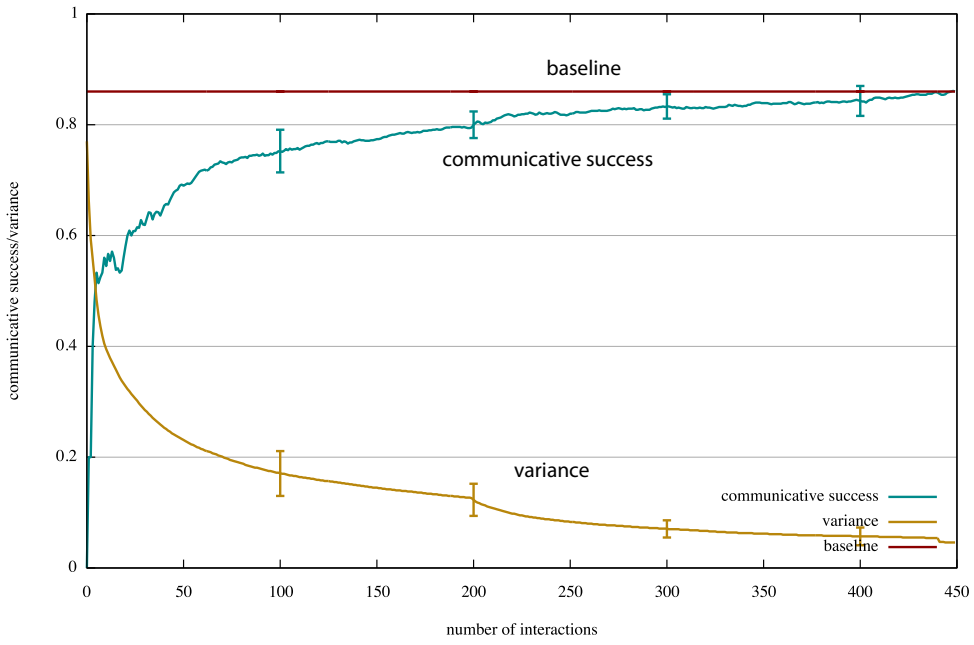



Figure 6.19: Learning scalable quantifiers. The graphs shows 6 trials of 450 interactions for a community of 5 agents. The graph shows that the communicative success increases and reaches the previously established baseline.

### 6.4.3 Formation experiment

The self-organization experiment for the scalable quantifier language strategy is identical to the one for absolute quantifiers. In this section we show that if agents are endowed with a minimal set of invention, learning and elimination operators, they will bootstrap an optimal (i.e., satisficing) quantifier system.

As in the previous experiment (see section 6.3), we bias agents towards a simple system of only two quantifers. Every newly invented quantifier is provided with a confidence score that reflects how well the quantifier performs. If that score drops below a specific threshold, the quantifier is removed from the ontology. Next to inventing new quantifiers, they continuously search to improve existing quantifiers.

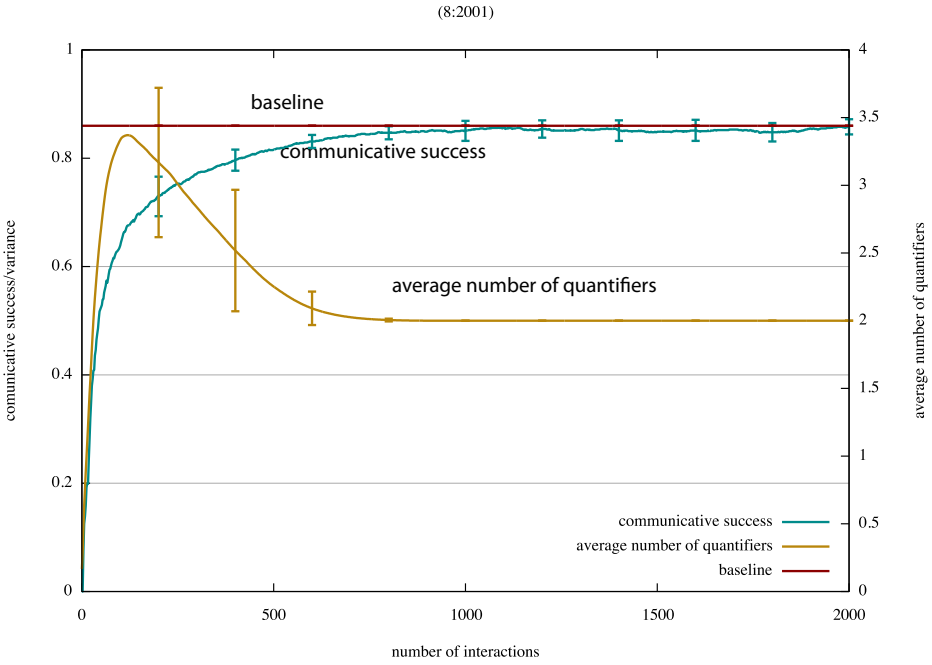



Figure 6.20: Self-orgaization of scalable quantifiers. The graphs shows 8 trials of 2000 interactions for a community of 5 agents. The graphs shows an increase of communicative success that converges on the baseline success after 800 interactions. We see an initial overshoot of lexicon size. After about 150 interactions, the number of quantifying expressions starts to decrease until after 800 interactions the agents converge on a set of two optimal quantifiers.

Figure 6.20 shows the results of a language-formation experiment. The average communicative success reaches the baseline. The baseline was established using optimally defined quantifiers, Thus the agents find a quantifier system that is optimal for scalable quantifiers.

Previously we have shown that a system based on scalable quantifiers performs better in this particular environment than a absolute-quantifier system. The result of this experiment shows that the agents can bootstrap a scalable-quantifier system that exploits this advantage. In the following section we build upon these findings by showing that as a consequence, when left to their owen devices, the robotic agents will favor a scalable- over an absolute-quantifier system.

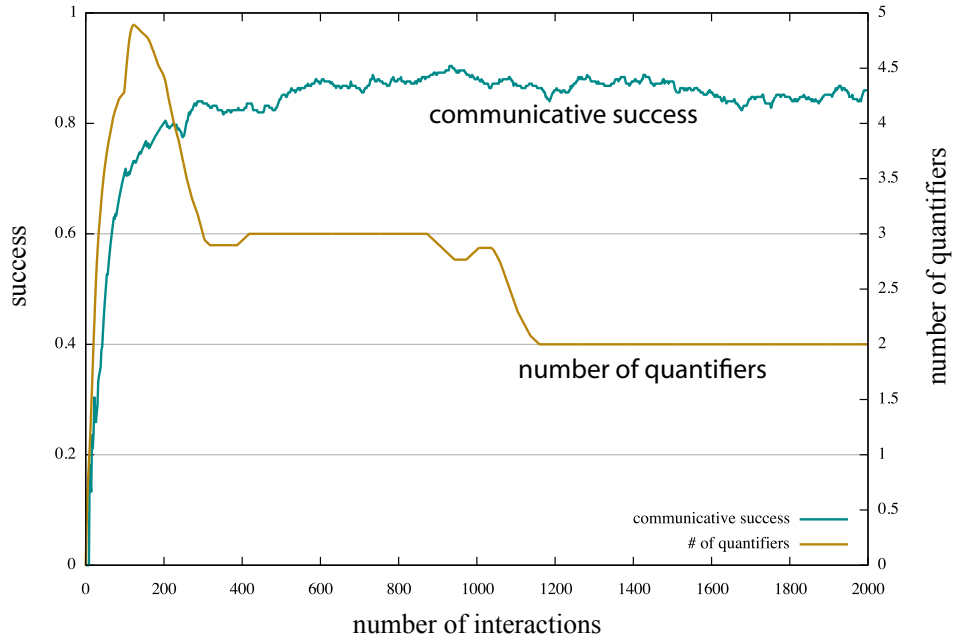
## 6.5 Experiment 3: Strategy competition

The above experiments considered the absolute and scalable language strategies in isolation. They show that for both both strategies a community of agents can self-organize a quantifier system using the same set of general learning and formation operations. Here we will show the results of putting both strategies in competition.

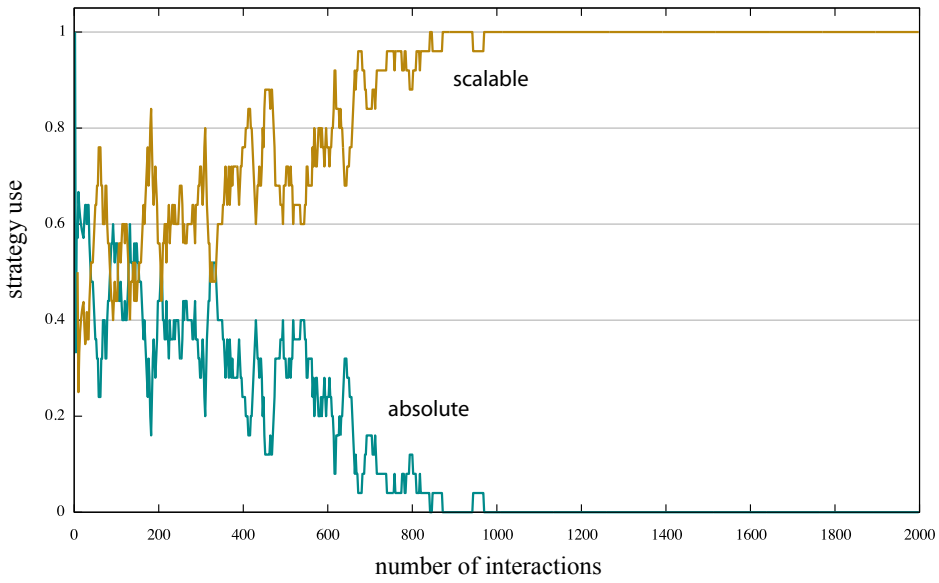
The experimental setup is identical to the formation experiments in section 6.3.6 and 6.4.3. The agents are provided with operations for learning and expansion of the language system. Every quantifier contains a confidence score that reflects its communicative performance. If the score of a specific quantifier is 0 it is removed. In order to keep the experiments throughout the chapter comparable, the agents are biased towards learning a system of two quantifiers. We consider two different environmental conditions. The first (*random*) condition presents the agents with a random set of scenes from the data set. In the second (*monotonic*) condition we ensure that the agents can alway discriminate the focus scene in terms of number of blocks, thus taking away the necessity to consider the number of boxes. On the basis of the previous experiments, we hypothesize that the random condition will give rise to a clear advantage for scalable determiners, the reason being that the scenes from the data set prototypically contain more blocks than boxes, thus making scalable quantifiers more useful than absolute ones.

Figure 6.21 shows the results of the experiment. Figures 6.21(a) and (b) show different plots for the same example run of the *random* condition. We see that the agents converge on a scalable quantifiers system in this scenario. Of course, depending on the run, it is possible for the the agents to converge on different strategies: only scalable quantifiers, only absolute quantifiers or a mixture of both. Figure 6.22, shows the average outcome of different systems over 15 runs for the two conditions (random and monotonic). We see that for the random conditions, the community of agents almost always converges on a scalable language strategy. For the monotonic condition there is no clear preference for one strategy over the other.

These results are in line with the earlier findings in this chapter. The baseline experiment showed that for the random condition there is a clear advantage for scalable quantifiers. The formation experiment in the previous section shows that agents are capable of self-organizing such a scalable quantifier system. Here




(a) communicative success of quantifiers




(b) strategy use

Figure 6.21: The results of the strategy competition experiment. Graphs (a) and (b) show different plots for the same run. Graph (a) shows the convergence of communicative success and graph (b) shows the strategy that is being used by the agents. In this particular run the agents converge on a quantifier system using the scalable language strategy. Initially they use both strategies, but at around interaction 1000 the last absolute quantifier dies out.

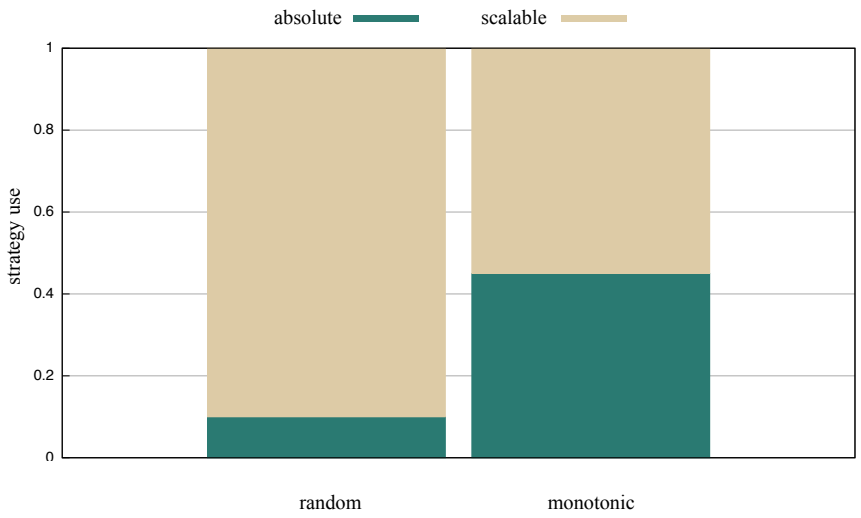



Figure 6.22: Graph (c) shows the result of 15 runs for two different conditions: random and monotonic. We see that for random scenes the agents almost always converge on a scalable strategy. For monotonic scenes the choice of strategy is random.

we show that the general selectionistic mechanisms exploit this advantage and will lead agents to develop a scalable quantifier system. These result confirm the hypothesis: The structure of the environment and the capacity of agents to perceive this structure can by itself explain the choice for a scalable quantifier system.

## 6.6 Conclusion

The present chapter demonstrates how a community of agents can self-organize a language system for quantificational expressions. Through a series of experiments we verified the hypothesis that predictability within a given environment is required to explain the need for context-dependent quantifiers.

The first experiment shows how agents can self-organize a quantifier system based on an absolute quantification language strategy. Such a strategy works well so long as the various types of obects do not show prototypicality effects regarding their numerosity. The second experiment shows how agents can self-organize a quantifier system based on a scalable quantification language strategy. For simple worlds that do not exhibit any structural systematicity with respect to the amount of objects of a specific type, such a strategy does not differ from absolute quantifiers. However, in more realistic situations, the advantage of scalable quantifiers becomes apparent. In the final experiment we allow the agents

to draw from both strategies in the development of a quantifier system. This experiment shows that the preference of a strategy depends on the predictability of the distribution of objects in the visual context. A high level of predictability will lead our agents to favor a scalable- over an absolute-quantifier system. The mechanisms proposed in this chapter are based on the general principles used throughout this book.

While the convexity constraint is a necessary (cognitive) condition, the main focus of this chapter is on the environmental constraints. Needless to say, in practice, both cognitive and environmental constraints influence the way a language develops. This experiment shows that the environmental constraints alone are sufficient to explain the tendency of languages towards scalable quantifiers. A next interesting step would be to look at the role of cognitive constraints (such as subitizing effects).

Another interesting next step would be to extend the current experiments beyond the domain of quantification. The principle of scalability is far from being exclusive to the domain of quantification. Adjective such as *far* and *near* and *big* and *small* show the same kind of scalability in their respective domains. In fact, there are languages such as Malagasy that use the same word for the quantifier *many* and the adjective *big* (von Fintel and Matthewson, 2007).

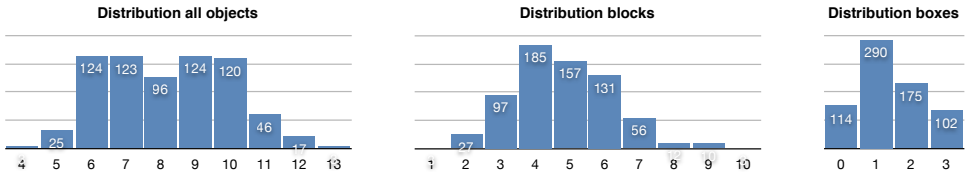
## Acknowledgments

The research reported in this chapter was conducted at the Sony Computer Science Laboratory and the Barcelona "Social Brain" Chair at the Universitat de Barcelona, funded from the FP7 EU project ALEAR. We would also like to thank Oscar Vilarroya, Mar Garachana, Jordi Fauquet, Javi Valenzuela, Luc Steels, and Emi Castaño for their input. And, we would like to thank Michael Spranger for gathering the raw robot data.

# 6.A Data

We use prerecorded data from robotic interactions. The data itself is not publicly available, but this section describes the relevant statistics.

The data set we use contains a total number of 681 scenes. Every scene contains a number of objects that are represented as a list of continues feature values (e.g. color channels, estimated position, estimated height, etc...). The precise features are not relevant for this chapter. We are mainly interested in the number of instances of a specific object type in a scene. Figure 6.23, shows the distributions of the two object types of interest: blocks and boxes.




(a) Total (avg = 8.2, stdev = 1.8) (b) Blocks (avg = 4.8, stdev = 1.5) (c) Boxes (avg = 1.4, stdev = 0.9)

Figure 6.23: The distribution of objects in our data set.

The robots have to discriminate scenes in terms of the number of objects of a specific type. The communicative success in doing so, depends on the precis quantifiers and can not be established *a priori*. We can, however, establish a theoretical upper limit. If the focus scene contains the same number of boxes and blocks as the alternative, then it will not be impossible to successfully discriminate the focus scene.

With a total of 681 there are 463761 possible combinations of scenes. The table in Figure 6.24 shows the number of scenes pairs that can discriminated in terms of number of blocks, boxes or either.

| object type | number of pairs | proportion |
|---|---|---|
| boxes | 325636 | 0.70 |
| blocks | 374190 | 0.81 |
| boxes or blocks | 435038 | 0.94 |

Figure 6.24: The distribution of objects in our data set.

In theory this means that with an unbounded number of quantifiers, the agents could achieve communicative success in 94% of the cases. However, in practice, even with an unbounded number of quantifiers, this practical limit will not be reached due to noisy perception of the robots.

# Chapter 7

# Size Matters: Adjectival Origins

Gradable quantifiers (e.g., *many* and *few*) have a dual nature: They can behave both as quantifiers and adjectives. Using a framework in which agents can self-organize a language, I examine the hypothesis that their adjectival use can be explained by the cognitive overlap between these quantifiers and adjectives such as *big* and *small*. This chapter is a version of a forthcoming publication: Pauw (2013c).

## 7.1 Introduction

Words like *many* and *few* are syntactic hybrids: though traditionally analyzed as quantifiers ("many of the houses"), they can also behave like gradable adjectives ("few/fewer houses"). In fact, such terms pattern syntactically and semantically with both quantifiers and adjectives (see Solt, 2009, for an extensive analysis of these syntactic patterns). Why aren't they confined to one grammatical class? What is the cognitive basis for their dual behavior? And how might such conceptual and linguistic duality have evolved?

Cross-linguistic properties of these terms (henceforth, *gradable quantifiers*) suggest close ties between the functions of quantification and predication. Some languages have no separate grammatical class for expressing number, but instead use size-modifying adjectives, as in the extension of the Pirahã predicate *hi* 'small' to indicate 'a small number' (Everett, 2005). Historical evidence also suggests that gradable quantifiers typically derive from adjectives, as illustrated by the quantifier *few*, based on the Old English adjective *feawe* (Solt, 2009).

This chapter explores the hypothesis that the duality of gradable quantifiers has its roots in the close cognitive relationship between size and number. Judgments of size (underlying modifiers such as *big* and *small*) depend on perceptual features of objects (or sets of objects) in the environment. Judgments of approximate number (underlying terms like *few* and *many*) exploit a combination of spatial features that apply exclusively to sets of objects, such as their size and

density (Durgin, 1995). This cognitive overlap between the concepts of size and number may account for the duality observed in gradable quantifiers: the dependence on size motivates their adjectival uses, while their application to sets of objects motivates their quantificational uses.

In the next section of this chapter I introduce a computational model that captures the insight above within the evolutionary language games framework (Steels, 2012a), in which robotic agents self-organize the means for describing objects (or in this case, groups of objects) in their perceived environment. Within this framework, agents are encouraged to develop a system of gradable quantifiers. The following section of this chapter, introduces two innovation strategies: Agents can either develop new quantifiers from scratch or they can develop quantifiers by extending the meaning of cognitively similar lexical items (e.g., gradable adjectives). In the last part of the chapter it is shown that when left to their own devices, the agents will prefer the latter strategy and therefor converge on a set of quantifiers that find their origins in adjectives.

## 7.2 Robotic Language Games

The experiments we describe are based on communicative interactions between humanoid robots (see (Steels, 2012a) for specific details). Figure 7.1 shows an example scene with two robotic agents interacting in a shared environment. We use a setup that has been used in many other experiments (Spranger, 2012; Pauw and Hilfery, 2012; Pauw and Spranger, 2010). Each robot perceives the world through its own on-board sensors, e.g., a camera and proprioceptive sensors. From this multimodal sensory input, the robots build a world model, which reflects the robot's current belief about the state of the environment.

Conducting experiments with actual robots is time consuming. I therefore reuse recorded data from actual robotic interactions to speed up our experiments while at the same time preserving the realism of physical robot interactions. Every scene contains two robots. The recorded data of the scene comprises the world models that both robots have built from their visual input using standard machine vision algorithms. In the world model of a scene, every object is represented as a list of features such as their width, height, color and position. (See, for example, Figure 7.2.)

Using this data, we let a community of agents play *language games* (Steels, 2012a). The type of language game the agents play depends on the particular research question. For the purpose of this chapter we let them play *Multi-Word Guessing Games* (Steels, 2012b). This game uses the following script:

1. One of the agents takes the role of a speaker, the other takes the role of hearer.

2. The speaker has to pick an object or a group of objects as the *referent*. This

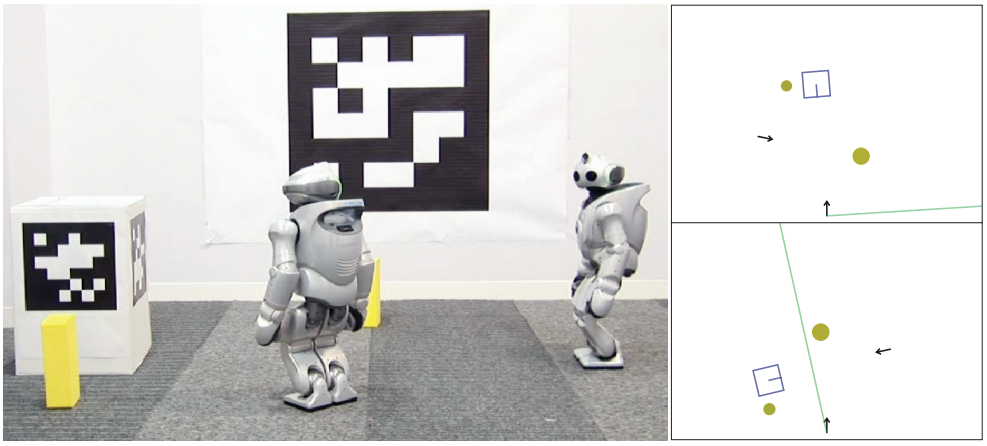


Figure 7.1: An example of a robotic interaction. Robots are placed in an office-like environment that contain different types of objects. This scene contains two yellow blocks, one box and two robots themselves. The world models of the robots are shown on the right. In the world models, the arrows represent the robots. The direction of the arrow marks the orientation of the robot. The circles represent the blocks and the blue square represents the box.

```
(object obj-228
 ((x 1868.59) (y 108.199) (z 0)
  (width 138.031) (length 131.142) (height 226.175)
  (average-y 85) (stdev-y 11.5768) (min-y 0) (max-y 112)
  (average-u 188) (stdev-u 14.1542) (min-u 0) (max-u 208)
  (average-v 122) (stdev-v 5.54406) (min-v 0) (max-v 128)))
```

Figure 7.2: An example representation of a block. Objects do not come pre-classified or predicated in any way. They are a list of continuous features. For this particular example we see its euclidean coordinates (in $mm$), its dimensions (in $mm$), and its color coordinates (in yuv color space).

is done randomly.

3. The speaker has to find an utterance that most clearly singles out the referent.

4. (a) If the speaker was capable of finding an utterance, the hearer has to interpret the utterance and point to the object or group of objects that he thinks that speaker intended.

   (b) If the speaker could not find an utterance, the game is considered a failure and ends here.

5. (a) If the hearer points correctly, the game is considered a success and ends here.

   (b) Otherwise the game is considered a failure and the speaker will point at the intended referent (thus creating a learning opportunity for the hearer).

So, this game can either end in a success (5a) or in a failure (4b and 5b). In the case of failure, the agents are provided with two *repair operators*, a *learning* and a *innovation* operator. The learning operator is used case of a failure on the hearer side (5b). The innovation operators is used in the case of a failure on the speaker side (4b). This operator extends the language of the speaker. Together these operators allow us to look at how a language can gradually change over time to incorporate novel communicative needs.

## 7.3 Model

Before looking at the experimental data, we need to have a more detailed look at the communicative machinery of the agents. The experiments reported in this chapter are certainly not built from scratch. They rely on mechanisms that have been developed and tested in other related experiments (Spranger et al., 2010b; Pauw and Spranger, 2010). The model that is described in this section is an extension of an existing model that has been used to study spatial language (Spranger, 2012) and quantification (Pauw and Hilfery, 2012).

For conceptualization this model uses a degree-based semantics that is built on top of a prototype system (Rosch et al., 2004; Lakoff, 1987). This combination of prototype theory and degree-based semantics has proven particularly successful in dealing with the problems that are inherent to conceptualizing real-world perception. (See Spranger and Pauw (2012); Pauw and Spranger (2012) for an in depth discussion.)

This semantics is implemented in a formalism called Incremental Recruitment Language (IRL) (Spranger et al., 2012b; Van Den Broeck, 2008; Steels, 2000b), a procedural semantics that is designed for semantic planning in robotic language

games. For parsing and production of language the agents rely on Fluid Construction Grammar (FCG) (Steels and De Beule, 2006). IRL and FCG have been co-developed specially for the kind of grounded language games as described in this chapter. The rest of this section discusses the different technical aspects of this model in more detail.

### 7.3.1 Conceptual system - IRL

In the present scenario, the meaning of language is grounded in the real world. That is to say, the robotic interactions rely on data from the vision system of these robots. This data does not come pre-classified, but rather every entity in the environment is represented as a list of continuous features. (Figure 7.2 shows an example representation of a block in the world model of a robot.) The task of the conceptual system is to bridge the gap between the continuous perception of the real world and the discrete conceptual world.

#### Prototype Theory

A combination of IRL and prototype theory has proven very successful in conceptualizing real-world perception. In prototype theory, every concept is represented by a *prototype*. A prototype is a prototypical value (or vector) in the relevant domain. For example, the concept *big* is defined over the surface area of an entity. The closer the surface area is to the prototypical value for *big*, the more similar it is to the concept *big*.

This intuition of similarity is captured by a similarity function. The similarity function takes a prototype and an entity and returns a confidence score between 0 and 1. If the score is 0, the entity is not similar at all to the prototype, if it is 1, the entity is precisely like the prototype. Consider for example the following similarity function for area prototypes:

$$S_{area}(P, O) = e^{\frac{1}{\sigma_P}|\text{area}(P)-\text{area}(O)|} \tag{7.1}$$

This is a typical similarity function in prototype theory. In this case it is defined over the feature *area*, but similar functions can be defined over any feature or set of features. The prototype defines a prototypical value for the relevant feature (area($P$)) and a rate at which the confidence score decreases with the distance to this value ($\sigma_P$).

For gradable modifiers such as *big* and *small*, however, this sketches a slightly oversimplified picture. These modifiers require a comparison to some context-dependent norm. In Pauw and Hilfery (2012) it is shown in detail how the notions of prototype- and exemplar-based models can be adapted to model this norm dependence. For every type of object, the agents know what a normal amount is

(the average size of all objects of the specific object type)[1]. The prototypes are scaled with respect to this amount. On top of that, the prototypes have an open side (i.e., on one side of the prototype, the score is always 1). (See Figure 7.3 for an example of the scaled prototypes for the object type block.)

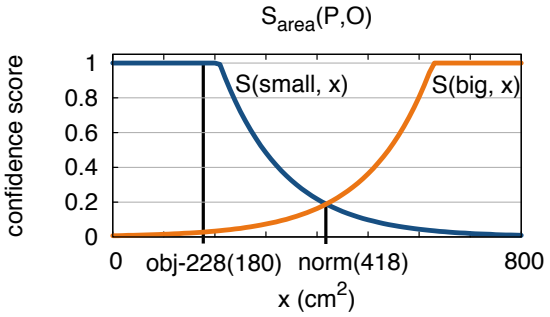



Figure 7.3: The area similarity function for blocks. The average block size is $418cm^2$. This is the norm. Any block that is bigger than that norm is gets higher confidence score for the *big*-prototype, any block that is smaller gets a higher confidence score for the *small*-prototype. The prototypes have an open side (i.e., on one side of the prototype, the score is always 1).

If we want to know if `obj-228` in Figure 7.2 is a small or a big block, we simply take its surface area and compare it to the *small* prototype using the above similarity function. The surface area of `obj-228` is $181cm^2$ (width = $138.031mm$ and length = $131.142mm$). If we enter this value in the similarity function above together with the *small* prototype we get a similarity value of 1. For *big* the similarity value is 0.03. So, the least we can say is that *small* is a much better way to describe object `obj-228` than *big* is.

### Composition

Every concept in the agents' ontology is defined in the same way, using prototypes and similarity functions. And, if agents would only need one concept to describe an object this would be the end of the story. Most utterances, however, involve more than one concept. Take for example, the utterance "big block", this contains both the concept *big* and *block*. We need some mechanism to combine both concepts. This is done by first computing the similarity scores for both concepts individually and than using a mixture function that computes a total score.

[1]It should be noted that the norm has different definition here than in Chapter 6. In Chapter 6 norm is defined as the maximal observed amount. Here, the norm is taken to be an average amount. The difference is mainly a technical design choice. The reason to change the definition is to be in line with other semantic theories (e.g., Fernando and Kamp, 1996).

There are many possible ways to define such a mixture function (e.g., the product of the two scores, or their average) and in studies of degree-based semantics, this is subject to much debate (van Rooij, 2011). For this experiment I have adopted Zadeh's definition for conjunction by using the minimum of the scores (Zadeh, 1965). All philosophical ramifications aside, there is a very practical reason to use the minimum as mixture function: The scores are likely to get lower when there are more concepts involved, thus discouraging the agents to come up with overly long and convoluted descriptions.

In sum, a concept (using prototypes and similarity functions) assigns a score to every entity in the environment. Using the minimum as a mixture function the scores of multiple concepts can be combined. So, when interpreted, an utterance like "big block" assigns a score to every entity that reflects how well this entity is described by the utterance.

The question remains: How are these scores being used in communication? The goal for the hearer is to point at the referent he thinks the speaker intended. So the hearer can just simply take the entity that has the highest score. For the speaker the story is slightly more complicated: For conceptualization, IRL has a search mechanism that finds a set of prototypes that together best describe the referent. In order for a description to be good, it should not only have a high score for the referent, but at the same time a low score for all other entities. IRL finds the set of prototypes that together maximize the difference in score between the referent and all other entities, a process called *contrast maximization* (see Chapter 3 of this thesis for more information on the search mechanism and Chapter 4 for an extensive study of the principle of contrast maximization in real-world perception).

### Quantifiers

The mechanism described above shows how concepts in the form of prototypes can be used and combined to find descriptions of single objects. However, this chapter focuses on quantifiers and quantifiers typically do not describe single objects, but sets of objects. So, how are sets of objects modeled? And, how are quantifiers used to describe them?

For the experiments described in this chapter, it is assumed that there is a fixed set of object groups in the world model of the agent. So, all the groups the agents could talk about are know before any communication has taken place. The groups are computed directly from the visual data using a standard clustering algorithm called agglomerative clustering (Mitchell, 1997). The algorithm creates a hierarchy of groups based on their spatial arrangement. Figure 7.4 shows an example scene containing object groups that were created using this agglomerative clustering approach. The amount of groups that is created depends on the granularity of the algorithm, in this example it happens to create three groups.

More important than how the object groups are computed is their represen-

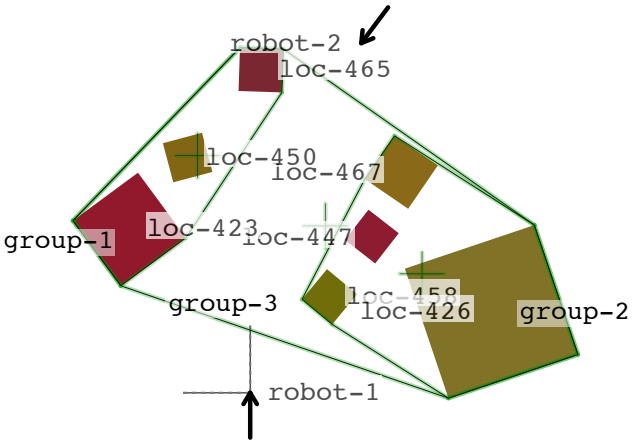
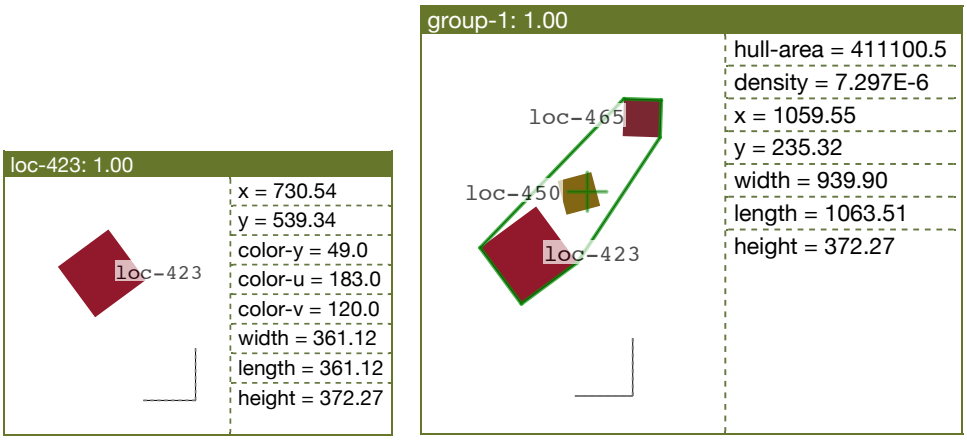



Figure 7.4: Top down view of an example scene. The arrows represent the speaker (*robot-1*) and the hearer (*robot-2*). The scene contains seven blocks (*loc*-x) and three object groups: *group-1*, containing the three leftmost blocks; *group-2*, containing the four rightmost blocks; and *group-3*, containing all blocks in the scene.

tation. Objects groups are represented in exactly the same way as single objects; i.e., a list of features. Most of the features that object groups have are the same as those of single objects (such as width, height, position, color parameters, ...). So, without any change to model, the agents can already use the concepts that are used for single objects to describe groups of objects, using utterances such as "the big [group of] objects", or "the [group of] objects to the left".

But, the objects groups, have an additional feature that single objects do not have: the density of the object group (the total surface area of the constituents divided by the surface area of the entire group). And, although the agents already have means to describe groups of objects without taking this feature into account, the agents can come across situations where it is useful to have means to take this density feature into account. And, as we will see in a moment, it is precisely this density feature that is being used by (gradable) quantifiers.

In order to fit the definition of the quantifiers *many* and *few* within the prototype framework, Pauw and Hilfery (2012) employ a exemplar-based approach, where the similarity between the cardinality of an object group and the concept is determined. There is a problem with this approach however: It requires that the agents know the precise cardinality of object groups. From a cognitive point of view this is not very plausible. When we use an utterance like "many blocks", we do not explicitly count the number of relevant objects, but we estimate the number of object. The cognitive mechanism that is responsible for number estimation is often referred to as the *Approximate Number System* (ANS) (Halberda and Feigenson, 2008). The precise working of the ANS is subject to much debate,



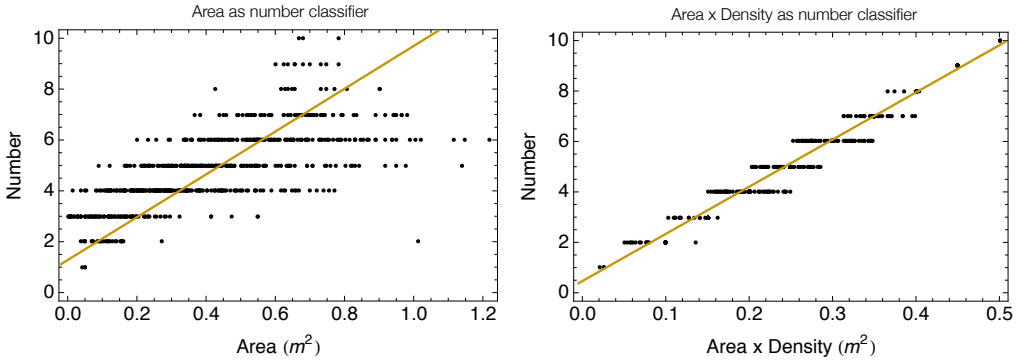

(a) Block (b) Object Group

Figure 7.5: Both single objects and groups of objects are represented as list of features. Most prototypes that apply to single objects can just as well be used for object groups.

but there seem to be two dominant features that influence the estimation of the amount of objects in a group: the *size* (area/volume) that the group of objects as a whole occupies (Hormann, 1983; Newstead and Coventry, 2000) and the *density* of the objects in the group (Durgin, 1995). (See Chapter 2 of this thesis for an in-depth discussion.)

In fact, when we look at the robot data, we can see that these features correlate very strongly with the number of objects in that group. Consider Figure 7.6(a). This figure shows both the surface area and number of objects for every possible group in all recorded scenes. There is a strong correlation between both variables ($\rho = 0.81$), making surface area a good classifier for number. But, we can do better: Figure 7.6(b), shows that, in line with the observation above, if we also take the density into account, we get an even better classifier. The density of an object here is computed by dividing the total surface area of all objects by the surface area of the group as a whole (including the space between the objects). By multiplying this density with the surface area of the group we get a value that correlates very strongly with the number of the group ($\rho = 0.96$).

Based on these observations, we can define the prototypes for the gradable quantifiers *many* and *few*. These prototypes do not have one dimension like *big* and *small*, but are defined over two dimensions: surface area and density. The similarity function for the Approximate Number System ($S_{\text{ANS}}$) then also has to take these two dimensions into account. The function $S_{\text{ANS}}$ is essentially an extension of $S_{\text{area}}$ that was used for the concepts *big* and *small*, but now also




(a) Area ($\rho = 0.81$) (b) Area×density ($\rho = 0.96$)

Figure 7.6: The correlations between number of objects, area and density. These measures are taken over all 900 recorded scenes. Every point represents a group of objects. The y-axis shows the number of objects in the group the x-axes shows its (a) area and (b) area times density. Both measures correlate with number, but the latter more strongly.

takes density into account:

$$S_{\text{ANS}}(P, O) = e^{\frac{1}{\sigma_p}|\text{area}(P)\cdot\text{density}(P) - \text{area}(O)\cdot\text{density}(O)|} \tag{7.2}$$

In sum, in this model, object groups are modeled in the same way as single objects. This means that they can be described using the same prototypes as used for describing single objects. However, object groups have the additional density feature that can be used to define concepts that exclusively apply to these object groups. The gradable quantifiers *many* and *few* are examples of such concepts. They extend the size-concepts *big* and *small* by taking the density feature into account.

It should be mentioned that this model of quantification is very different from most existing approaches. In most existing approaches quantifiers have a different functional status than adjectives. For example, in Generalized Quantifier Theory(Barwise and Cooper, 1981) adjectives are simple predicates and quantifiers are relations between predicates. Also earlier similar computational experiments assumed that quantifiers and adjectives are functionally different (Pauw and Hilfery, 2012; Spranger and Pauw, 2012). For the present experiment however, it is essential to let go of this functional difference because this allows gradable quantifiers to emerge as adjectives.

### 7.3.2 Grammar - FCG

The previous section describes how the agents can conceptualize the world using prototypes and similarity functions. Here we will look at how these concepts are communicated. In the experiments to follow the agents can use common nouns (e.g., "big blocks") and noun phrases (e.g., "two big blocks") to describe the (groups of) objects in their environment. The agents are provided with a basic context-free grammar that allows them to recursively build common nouns and noun phrases. The grammar is implemented in Fluid Construction Grammar (FCG) (Steels and De Beule, 2006), a construction grammar that links the syntactic structure of an utterance directly to the semantic operations of the agent. Figure 7.7 shows the syntactic structure of the utterance "two big blocks".

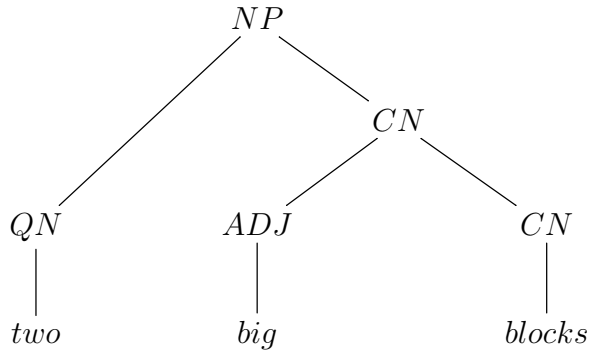
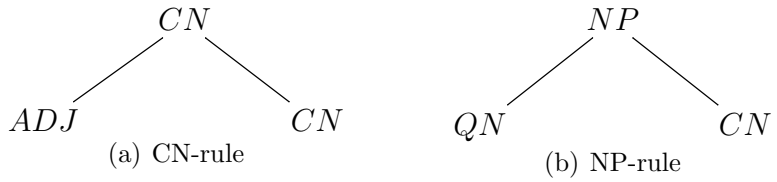



Figure 7.7: Syntactic structure of "two big blocks". FCG combines the adjective (ADJ) 'big' and the common noun (CN) 'blocks' into the CN "big blocks", which together with the quantifier (QN) 'two' forms the noun phrase (NP) "two big blocks".

The grammar consists of lexical rules that link the labels to the concepts (prototypes) and their respective grammatical category. For example, there is a lexical rule which links the label 'big' to the category adjective and the *big*-prototype. There are three grammatical categories for lexical items: adjectives (ADJ ) such as *big* and *left*, quantifiers (QN) such as *two* and *three*, and common nouns (CN) such as *block* and *box*.

Furthermore, the grammar contains two grammatical constructions that recursively combine the lexical items into more complex CN's and NP's (as shown in Figure 7.8). Note that these constructions are defined in a construction grammar, so they do not only define the syntactic patterns, but they also have semantic contents (Goldberg, 1995). They instruct the agent how the concepts of its constituents should be interpreted.

The CN rule instructs the agents to apply both the prototypes of the ADJ




(a) CN-rule (b) NP-rule

Figure 7.8: Grammatical constructions. The CN rule The NP rule combines a QN and a CN into a NP.

and of the CN to the context and use the mixture function (as defined above) to combine the two. For example, in Figure 7.7, the CN instructs the agent to apply both the *block*-prototype and the *big*-prototype to the context and combine the results. The NP rule does the same thing, but additionally instructs the agents to ignore single objects and only consider groups (sets) of objects. In theory the length of the CN's that this grammar produces is unbounded. However, the amount of used modifiers are limited by the conceptual system. So, in practice, the agents will mostly construct noun phrases containing only one or two modifiers.

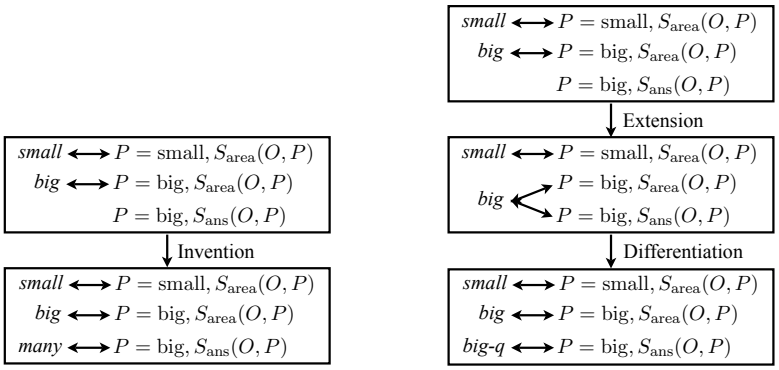
## 7.4 Language Innovation

Now we can turn to the main topic of this chapter: The evolution of gradable quantifiers. We provide the agents with lexical items to express spatial relations (*left*, *right*, *front*, *back*, *near* and *far*), size (*big* and *small*), object class (*block*, *box* and *robot*) and the grammatical rules to combine those items into nouns and noun phrases such as "block in front of you". We let the agents play language games as described above in which the agents have to describe (groups of) objects to each other. One of the salient properties of groups of objects can be the (approximate) number of objects it contains. Since the agents do not have any predefined vocabulary for expressing number they will have to develop this themselves.

To this end, I provided the agents with three operators: an innovation operator, which allows an agent to invent a new language item when required; an adoption operator, with which agents can learn the meaning of unfamiliar language items; and an alignment operator, which keeps track of the success of a language item.

### Innovation

The *innovation operator* is used when an agent wants to express a meaning for which it does not yet have a lexical item, so it has to establish a relation between a (new) lexical item and the concept that could previously not be expressed. This




(a) Invention Strategy (b) Extension Strategy

Figure 7.9: Two contrasting formation strategies. When the need arises to express a new meaning ($P = \text{big}, S_{\text{ans}}(O, P)$), the agents can either (a) create a new random lexical item for it (*many*), or (b) extend an existing lexical item (*big*). The latter strategy introduces homonyms. If the ambiguity that arises from such a homonym proves to be problematic, agents can invent a marker to differentiate (*-q*).

can be done in many ways. In most existing experiments that model language evolution, new language items are invented from scratch. I.e., as soon a novel concept needs to be expressed, the innovation operator randomly generates a label, which is than linked to that concept. Although fine for the purpose of those experiment, this innovation strategy is too much of a simplification for the present experiments. This chapter aims to test the idea that gradable quantifiers are derived from adjectives, this means that we cannot assume that they are invented from scratch. So, in the present chapter we compare two formation strategies (see also Figure 7.9):

1. The (traditional) invention strategy: Every time an agents wants to express a new meaning, a new random lexical items is created and mapped to this meaning.

2. The extension strategy: The agent tries to find existing lexical items that have a meaning that is *similar* to the new one. If it can find such a lexical item, it will extend its meaning rather than inventing a new lexical item. Remember that the meaning of a word is defined as a prototype and a similarity function. Thus, two meanings are identical when they have both the same prototype and similarity function. We say that two meanings are *similar* if they contain either the same prototype or the same similarity function.

   The extension strategy introduces an additional problem: It creates ambiguity. If a word has multiple meanings this can lead to communicative

failure. If such is the case, the speaker will notice this and he can decide to introduce a marker to differentiate the two interpretations.

### Adoption

Once an agent has created a new lexical item, this his to be learned by other agents in the community. This is taken care of by the *adoption operator*. This operator is used every time an agent hears an unfamiliar language item. In essence the adoption operator is the mirror-image of the innovation operator. The operator relies on a mechanism in IRL called *flexible interpretation*. When part of an utterance is missing, the agent can only derive part of the conceptual structure. The flexible interpretation mechanism can recover the missing part of the conceptual structure by trying out which combination of available operations and prototypes results in the discrimination of the referent, in the given context. The reconstructed part of the meaning is taken to be the meaning of the unfamiliar lexical item. Most of the time this operator creates many possible solutions.

The question is remains of how to pick one of those solutions. First of all, the amount possible solutions can be further limited. The adoption operator looks in the lexicon of the agents for similar lexical items. If such an item is found it is assumed that the new item was created by the extension strategy and therefore, the meaning of the new lexical item should be similar to the meaning of the similar lexical item. Secondly, if multiple solutions are found they can all be added to the lexicon. In such a case, the agents will have multiple language constructions that stand in competition.

### Alignment

Using the invention operator, a community of agents will introduce new lexical item. Most of the time, a community of agents comes up with multiple lexical items that describe the same concept. These language items are in competition with each other. This is where the *alignment operator* comes into play. The language items are all provided with a score between 0 and 1. When the items are newly created (either through innovation or adoption), the score is set to 0.5. After a successful interaction the operator increases the score of all the language constructions that were involved and, conversely, after every unsuccessful interaction the scores are decreased[2]. The scores are updated using the following update function:

$$S_{n+1}(l) = S_n(l) + \lambda S_n(l)(1 - S_n(l))$$

Where $S_n(l)$ is the original score and $S_{n+1}(l)$ is the updated score of lexical item $l$ and $\lambda$ is a rate at which a single update influences the score. For successful

[2]Note that the alignment operator in this chapter differs from the one described in Chapter 6. Here the alignment operator only adapts the score of the language item, whereas the alignment operator in Chapter 6 can also modify the underlying concept.

interactions $\lambda = 0.1$ for unsuccessful interactions $\lambda = -0.1$. The function is essentially a numerical approximation of an S-curve. This means that when the score gets closer to its extreme values (0 or 1), the rate at which the score changes gets lower. Update functions with this property have proven to be more stable than linearly updating scores for similar experiments. (I.e., a false positive or negative has less influence on the confidence score.)

So, when a community of agents is confronted with a novel problem (e.g., describing a group of objects in terms of its estimated number), these three operators assure that the community extends the existing language to accommodate this. The innovate operator creates new lexical items, and the adoption and alignment operator assure that the new items are spread through the community.

## 7.5 Experiments and Results

We can now look at how a community can use the described model and the innovation strategies to bootstrap a system of graded quantifiers. In what follows I will describe three experiments: A first baseline experiment establishes the expected communicative success of the quantification system. In this experiment the agents do not yet have to invent gradable quantifiers themselves. In a second experiment, the performance of both innovation strategies are tested individually. The last experiment finally shows that when the choice of strategy is left to the agent, they will be inclined to develop quantifiers that derive from adjectives.

### 7.5.1 Baseline

The first part of experimental section consists of a baseline experiment. Figure 7.9 describe different stages of development for both the extension and the invention strategy. Before looking at how agents can bootstrap a quantifier system, I implemented the language at each stage manually, considering four different stages of language development:

1. The agents have no gradable quantifiers. This is the initial state for both the invention and the extension strategy.

2. The agents are provided with size words that are extended to incorporate the ANS meaning. This is the second stage of the extension strategy.

3. The polysemy of the size markers is resolved by marking the different interpretations. This is the final stage of the extension strategy.

4. The estimate number is expressed using novel lexical items. This is the second and final stage of the invention strategy.

On top of these four different language conditions, there are two different environmental conditions: *Salient number*—the context and topic is chosen such that there is a guaranteed difference in the number of objects in the topic and any other group of objects in the context. In this condition, quantity always the most (and often, the only) salient feature of an object group; *No manipulation*—the topics and contexts are chosen completely at random from the data set. Figure 7.10 shows the average communicative success of 2000 interactions for each of these $2 \times 4$ different experimental conditions.

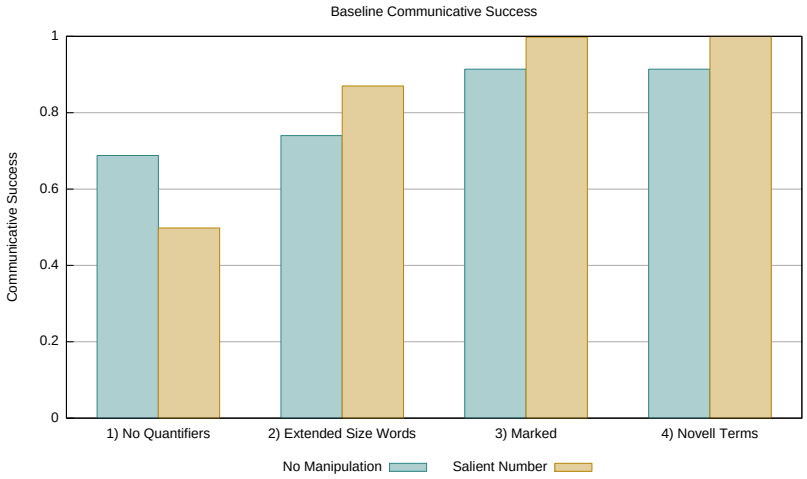



Figure 7.10: Results baseline, 2000 interactions, 2 environmental conditions, 4 linguistic stages. The height of the bar indicates the average communicative success over 2000 interactions.

The results show first of all that even without quantifiers, the agents can communicate. The number of objects is only one of the features they can use to describe a group of objects. Also, without quantifiers the agents are much less successful in the salient-number condition (50% success) then when there is no environmental manipulation (73%). Secondly, for both environmental conditions these results show that extending the size words strongly increases the communicative success (89% for the salient-number condition) and even more so when this difference is marked (100% for the salient-number condition). This effect is stronger for the salient-number environmental condition.

It should also be noted that there is no difference at all in communicative success between the third and fourth language condition. In other words, the end stages of both innovation strategies are equally successful (86% for the no-manipulation condition, 100% success for the salient-number condition). It is only the intermediary stage of the extension strategy which makes a difference.

### 7.5.2 Strategy Comparison

Now we can look at how agents can bootstrap a language using the proposed innovation strategies. In Figure 7.11, both the extension and invention strategies are tested individually. The Figure contains one a graph for both environmental manipulation: no manipulation and salient number. Both graphs show the results for both innovation strategies. For every condition I ran ten series of 1200 language game interactions. The graphs show the average progression of communicative success and lexicon size over those ten series.

These graphs show several things. To begin with, for both environmental conditions the community of agents reach the communicative success that was established in the baseline experiment (86% for the no-manipulation condition, 100% success for the salient-number condition). For the final communicative success it does not matter whether the agent employ the extension or invention strategy. Using either strategy they converge on the same communicative success.

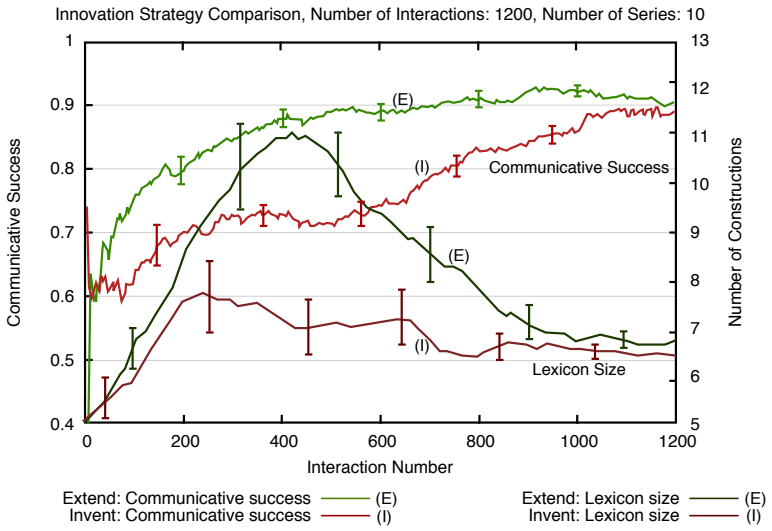
Second, the graphs show that there is an initial overshoot of the lexicon size. Initially all agents invent there own way of expressing quantity. Only when the lexicon becomes shared by the entire population (through the adoption operator) the unsuccessful lexical items gradually disappear and the language system converges on a smaller but more effective lexicon.

Of most interest is that, although the final success of both strategies is the same, there is a big difference in the rate at which this success is reached. Using the extension strategy a community of agents converges on a shared quantifier system much faster. For example in Figure 7.11(a), using the extension strategy the community of agents reach 80% communicative success after 200 interactions, whereas with the invention strategy this takes 800 interactions.

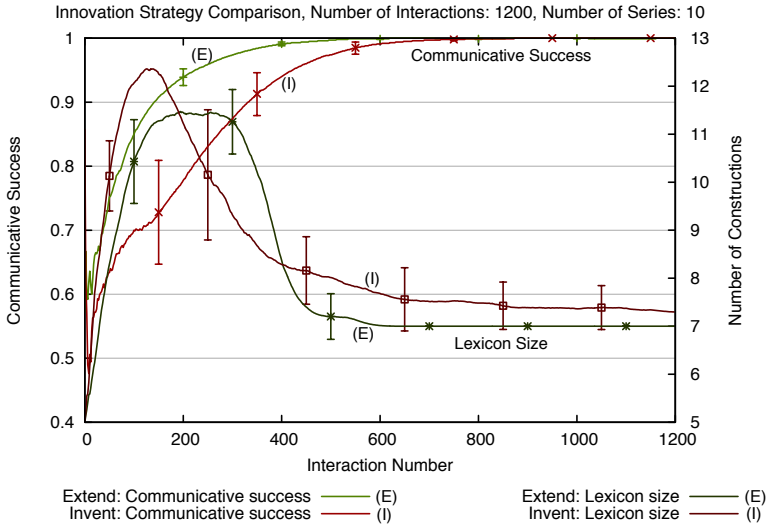
Finally, there is a clear difference between the two environmental conditions in this graph. Under the salient-number condition the agents start at only 50% success and reach 100% success in the end. This effect is much less spectacular without manipulation of the environment (73% to 86% communicative success). However, the advantage of the extension strategy become much more apparent for the no-manipulation environmental condition.

### 7.5.3 Strategy Competition

Figure 7.12 shows what happens if we leave the agent to his own devices in choosing a innovation strategy. Every time an agent needs to express a new meaning, it can choose to either follow the invention strategy and invent a completely new lexical item, or follow the extension strategy and adapt an existing lexical item. When agents randomly pick a strategy every time they require a new language item, they will wind up with a lexicon containing both gradable quantifiers that are derived from adjectives and quantifiers that are created from scratch. Figure 7.12(a) shows and example of an experiment where initially both types of quan-




(a) Strategy Comparison. No Manipulation




(b) Strategy Comparison. Salient Number

Figure 7.11: Results strategy comparison. These graphs show the average progression of the communicative success and lexicon size of the agents over 10 series of 1200 interactions. Both graph show the results of both the extension and invention strategy. Figure (a) shows the results for the no-manipulation condition and Figure (b) the results for the salient-number condition. The horizontal axis shows the number of interactions; left vertical axes, the communicative success; and the right vertical axis the lexicon size.

tifiers are invented, but the quantifiers that survive in the end have adjectival origins. Figure 7.12(b) shows a similar experiment, but here the community of agents converge on quantifiers that were created from scratch.

I have conducted this experiment many times, checking at the end of each series on which type of quantifiers the agents converged. The result strongly depends on the likelihood of either strategy being used. Figure 7.12(c) shows the likelihood that the converged quantifiers have adjectival origins as a function of the likelihood that the extension strategy is being used for innovation.

The graph shows no surprising results for the extreme cases: If the agents always use the extension innovation strategy the quantifier origins are always adjectival. And, conversely if the agents never use the extension strategy (in other words, always use the invention strategy) the resulting quantifiers are always created from scratch.

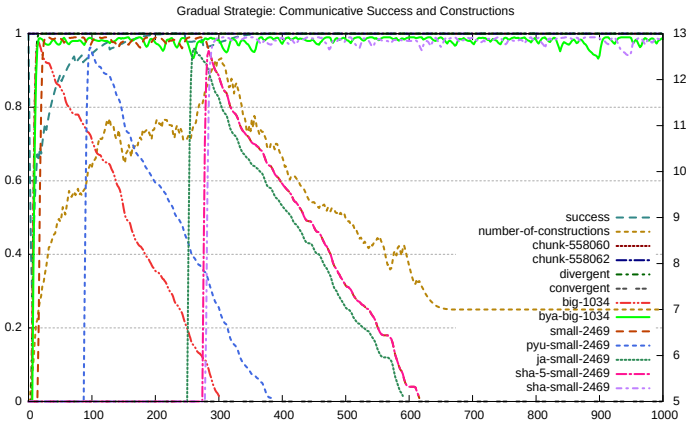
Of more interest is that Figure 7.12(c) shows an overall preference for quantifiers that have adjectival origins. At a likelihood of 50% for using the extension strategy, the chance that the agents converge on quantifiers with adjectival origins is between 70% and 85% (depending on the environmental manipulation). Overall, the likelihood of adjectival quantifiers is much higher than the likelihood of using the extension strategy (i.e., quantifiers that have adjectival origins).

In sum, both innovation strategies, extension and invention, permit a community of agents to develop a language containing gradable quantifiers. And for both strategies the resulting languages are equally apt in expressing number. However, the extension strategy exploits the cognitive similarity between size and quantifying terms. This results in an intermediary language stage in which the community of agents use size terms to express quantity. This intermediary stage makes it easier for novel terms to become shared in a population. So, leaving the choice of strategy up to the agents results in a preference for quantifying terms that are created by the extension strategy. In other words, all other things being equal, the community of agents has a strong preference for quantifiers with adjectival origins.

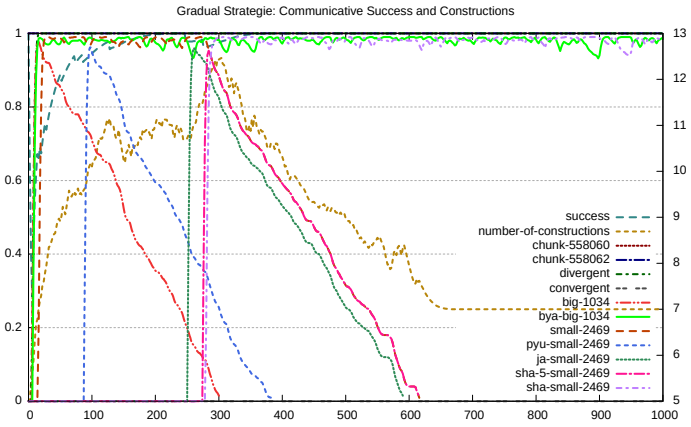
## 7.6 Conclusion

The present results provide a motivation for adjectival origins of gradable quantifiers. It was argued that these origins might be explained by a cognitive overlap between size terms and gradable quantifiers. I provided our agents with a cognitively plausible approximate number system and innovation strategies that can exploit the cognitive overlap for the generation of new lexical items. And it is shown that under these conditions, a community of agents is indeed inclined to derive gradable quantifiers from gradable adjectives.

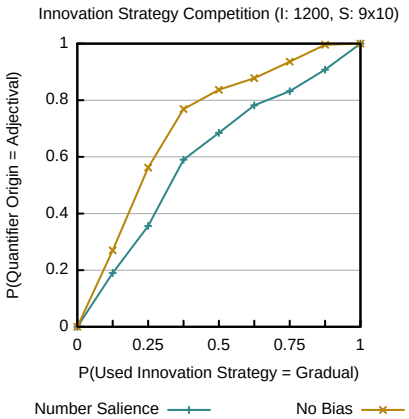
Of course, this is not the be-all and end-all answer to the duality of gradable quantifiers. Most importantly, this chapter only addresses part of the proposed




(a) Single experiment. Converged on quantifiers that were invented from scratch




(b) Single experiment. Converged on quantifiers with adjectival origins.




(c) Strategy Competition

Figure 7.12: Results strategy competition. Figure (b) shows an example of an experiment that results in graded quantifiers from scratch. In Figure (a) the agents converge on graded quantifiers with adjectival origins. Figure (c) shows the likelihood that the converged quantifiers have adjectival origins as a function of the likelihood that the extension strategy is being used for innovation. The result reflect the average convergence of 10 runs for every experimental condition.

puzzle. The chapter starts with pointing out the dual use of gradable quantifiers: Sometimes they act like adjectives and sometimes like quantifiers. This chapter only provides some insight on the adjectival origins. The question remains of why they can also occur as quantifiers. This could have to do with semantic overlap: Just as more canonical quantifiers, gradable quantifiers can only be applied to sets of objects, not to single objects. For now, I leave this as point of future investigation.

# Chapter 8

# More on Many and Few: Grammaticalization of Gradable Quantifiers

Modifiers such as *many* and *few* are known to be syntactic hybrids, acting alternately as quantifiers and adjectives. It has been argued that this duality in syntax is the result of grammaticalization, where these modifiers originate as adjectives and later become quantifiers. The present chapter describes an agent-based computational model that combines these linguistic insights with results from cognitive studies. Using this model it is shown that existing quantifiers can create attractor positions for other modifiers to grammaticalize into and that *many* and *few* follow this path in search of reducing cognitive effort. Thus, arguing that the shift from qualifying to quantifying expression has a cognitive motivation. This chapter is a version of a forthcoming publication: Pauw (2013a).

## 8.1 Introduction

Words such as *many*, *few*, *much* and *little* (henceforth *gradable quantifiers* ) are known to be syntactic hybrids: though traditionally analyzed as quantifiers, they can also act like adjectives (Solt, 2009). One of the main reasons for their classification as quantifiers is their position in a noun phrase: just like more canonical quantifiers, such as *all* or *three*, they cannot follow an adjective (e.g., we can say "many big houses", but not "big many houses"). However there are many constructions where gradable quantifiers behave like adjectives. Examples are "too many" and "fewer": we can say "too big" or "smaller", but not "too three" or "aller". Why do gradable quantifiers have such a fuzzy syntax? What is the cognitive basis for this syntactic duality?

It has been argued that the dual syntax of gradable quantifiers might be the result of a grammaticalization process where they originate as adjectives and later become quantifiers (Solt, 2009). In Chapter 7, I showed that gradable quantifiers

are indeed likely to emerge as adjectives due to their cognitive relation with size predicates such as *big* and *small* (perception of quantity relies on the same cognitive mechanisms as perception of size). Gradable quantifiers are not only cognitively related to size predicates but also to quantifiers such as *all* or *three* (they are used to describe sets of objects, not individual objects). In the current chapter I show that this cognitive relation will invite the (initially adjectival) gradable quantifiers to grammaticalize into quantifiers. So together with Pauw (2013b) this chapter shows that the syntactic duality of gradable quantifiers might reflect an underlying cognitive duality.

This chapter describes an *evolutionary linguistic* treatment of this grammaticalization process. In evolutionary linguistics, language is seen as a complex adaptive system that is subject to selectionist principles similar to those of biological evolution. The main agenda of this field is to identify the changes of language that led up to the current state of language and expose the cognitive mechanisms that are responsible for these changes. Lacking empirical data, computational modeling is the most common methodology in this field (See Kirby and Hurford, 2002; Steels, 2005). The objective of such computational experiments is to take changes observed by historical linguists and recreate them using (robotic) multi-agent simulations. This allows us to carefully study the cognitive processes that are required by the agents to produce the simulated language change. Of course it is not feasible to simulate the entire evolution of language from its genesis to the current state of all languages. Instead, it is common practice to focus on one specific aspect of language change and expose the cognitive mechanisms responsible for that particular change (see Bleys et al., 2009; van Trijp, 2012; Spranger, 2012; Pauw and Hilfery, 2012; Beuls and Steels, 2013, for recent examples).

As mentioned above, this chapter focuses on the grammaticalization path of gradable quantifiers from adjectives to quantifiers. It describes a series of experiments, in which a community of robotic agents is provided with a fully operational initial language that allows them to describe objects and sets of objects using nouns and noun phrases such as "two big blocks" or "many boxes". The gradable quantifiers *many* and *few* in this language are defined as adjectives. The agents are provided with a *reanalysis mechanism* that allows them to change the grammatical categories of lexical items (Bisang, 1998). It could, for example, change *many* into a quantifier.

The proposed reanalysis mechanism is a very general mechanism that could change the grammatical category of any lexical item into any category that has been provided by the grammar. However, in the experiments presented in this chapter, only gradable quantifiers grammaticalize into quantifiers, the grammatical categories of all other lexical items stay the same. The reason that only gradable quantifiers are affected by the reanalysis mechanism is due to the cognitive relation between gradable quantifiers and quantifiers such as *all* or *three*. Unlike adjectives such as *big* and *small*, these gradable quantifiers can only be

used to describe sets of objects. It is shown that this cognitive relation makes gradable quantifiers more prone to grammaticalization into quantifiers then other lexical items.

The remainder of the chapter is structured as follows: The next section discusses the computational framework and the implementation of the semantic and syntactic model. Sections 8.3 and 8.4 describes a series of experiments that show how languages in the different stages of the grammaticalization process perform. In a final experiment, described in Section 8.5, the agents are provided with the reanalysis mechanism, and it is shown that this indeed gives rise to the hypothesized grammaticalization path.

## 8.2 Experimental Framework

The experiments I describe in this chapter are based on communicative interactions between humanoid robots (see (Steels, 2012a) for specific details). Figure 8.1 shows an example scene with two robotic agents interacting in a shared environment. I use a setup that has been used in many other experiments (Spranger, 2012; Pauw and Hilfery, 2012; Pauw and Spranger, 2010). Each robot perceives the world through its own on-board sensors, e.g., a camera and proprioceptive sensors. From this multimodal sensory input, the robots build a world model, which reflects the robot's current belief about the state of the environment of gradable quantifiers from adjectives to quantifiers.

Conducting experiments with actual robots is time consuming. I therefore reuse recorded data from actual robotic interactions to speed up our experiments while at the same time preserving the realism of physical robot interactions. Every scene contains two robots. The recorded data of the scene comprises the world models that both robots have built from their visual input using standard machine vision algorithms. In the world model of a scene, every object is represented as a list of features such as their width, height, color and position.

Using this data, I let a community of agents play *language games* (Steels, 2012a). The type of language game the agents play depends on the particular research question. For the purpose of this chapter they play a *Multi-Word Guessing Game* (Steels, 2012b). In this game there are two agents, one of them takes the role as a speaker, the other as hearer. The speaker has to pick an object or a group of objects (the referent) and find an utterance to describe it to the hearer. The hearer in turn has to interpret this utterance and point at the referent he thinks the speaker intended. If the hearer points correctly, the game is a success. It is a failure otherwise. If the game is a failure, the speaker points to the referent to show the hearer what he meant.

When the speaker cannot find an appropriate utterance to describe the referent, or the hearer does not point correctly, the agent is provided with a set of *repair strategies* to resolve the situation. These repair strategies allow an agent

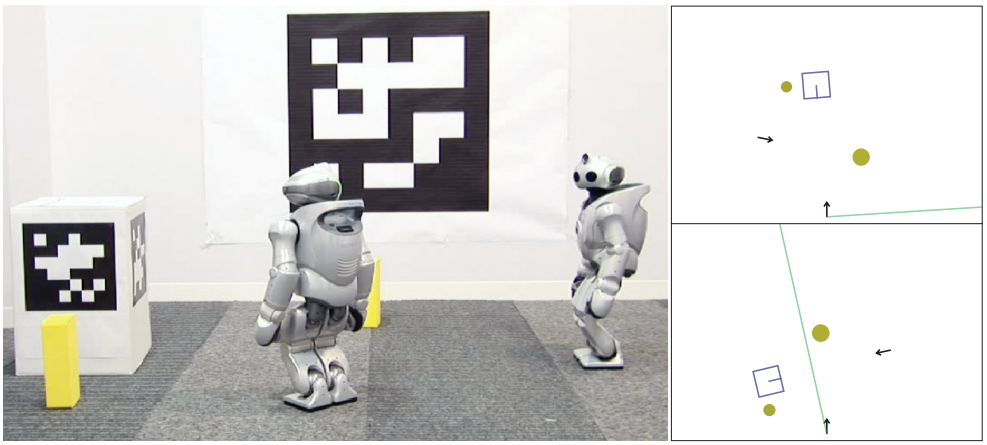


Figure 8.1: An example of a robotic interaction. Two robots, a speaker and a hearer, are placed in an environment containing different types of objects. This scene contains two yellow blocks, one box, and the two robots themselves. The world models of the robots are shown on the right. The world as perceived by the speaker is shown on top, the world model of the hearer is shown below. The arrows represent the robots. The direction of the arrow marks the orientation of the robot. They place themselves in the origin of their respective world models, looking in the direction of the x-axis. The circles represent the blocks and the blue square represents the box.

to adapt the language they use to the particular communicative needs of the situation. They are essential to this approach because they allow us to investigate language as an ever-adapting fluid system rather than a static given. In the case of the experiments presented in this chapter, the agents are provided with a repair strategy that allows them to reanalyze the grammatical categories of existing lexical items. In the remainder of this chapter I will show how this *reanalysis operator* gives rise to the grammaticalization of gradable quantifiers.

### 8.2.1 Model

Before looking at the experimental data, I will briefly discuss the communicative machinery of the agents. The experiments reported in this chapter are certainly not built from scratch. They rely on mechanisms that have been developed and tested in other related experiments (Spranger et al., 2010b; Pauw and Spranger, 2010). The model that is described in this section has previously been used in Pauw (2013b) and is an extension of an existing model that has been use to study spatial language (Spranger, 2012) and quantification (Pauw and Hilfery, 2012).

For conceptualization this model uses a degree-based semantics that is built on top of a prototype system (Rosch et al., 2004; Lakoff, 1987). This combination of prototype theory and degree-based semantics has proven particularly successful in dealing with the problems that are inherent to conceptualizing real-world perception (see Spranger and Pauw, 2012; Pauw and Spranger, 2012, for an in-depth discussion). This semantics is implemented in a formalism called Incremental Recruitment Language (IRL) (Spranger et al., 2012b; Van Den Broeck, 2008; Steels, 2000b), a procedural semantics that is designed for semantic planning in robotic language games. For parsing and production of language the agents rely on Fluid Construction Grammar (FCG) (Steels and De Beule, 2006). IRL and FCG have been co-developed specially for the kind of grounded language games as described in this chapter. The rest of this section discusses the different technical aspects of this model in more detail.

As mentioned in the introduction, the experiments reported do not study the evolution of language from its genesis. The experiments presuppose a language stage in which nouns, adjectives and quantifiers have developed into independent grammatical categories. This language allows the agents to use noun phrases (NP, e.g., "two big blocks") and common nouns (CN, e.g., "big blocks"). This language is an implementation of a basic context-free grammar in FCG. A CN consists of a zero or more adjectives (ADJ) followed by a noun. A NP consists of a quantifier (QN) followed by a CN.

Semantically, the function of a NP or CN is to assign a score to every object in the environment that reflects how well the object is described by the utterance. Every noun, adjective, and quantifier is associated with a prototype. IRL compares the prototype of a lexical item to every object in the world model of the agent, which results in a confidence score. When an utterance consists of multi-

ple lexical items these scores are combined by taking the minimum (see Zadeh, 1965; Pauw and Hilfery, 2012, for details). This establishes a confidence score for every object in the environment. During interpretation, IRL looks for the object that has the highest score for the given utterance. During language production (formation), IRL finds the utterance that has the highest score differential for the referent (i.e, the utterance that maximizes the score difference between the referent and all the other objects in the environment). Pauw (2013b) explains the prototype/degree-based approach in more detail and Spranger and Pauw (2012) discusses how this can be implemented in IRL.

It should be mentioned that the model of quantification in the present model is very different from most existing approaches. In most existing approaches quantifiers have a different functional status than adjectives. For example, in Generalized Quantifier Theory (Barwise and Cooper, 1981) adjectives are simple predicates, but quantifiers are relations between predicates. Also earlier similar computational experiments assumed that quantifiers and adjectives are functionally different (Pauw and Hilfery, 2012; Spranger and Pauw, 2012). For the present experiment however, it is essential to let go of this functional difference because this allows gradable quantifiers to emerge as adjectives (as shown in Pauw (2013b)). To be able to model quantifiers in the same way as adjectives, it is assumed that there is a fixed set of object groups in the world model of the agent. The groups are computed directly from the visual data using a standard clustering algorithm called agglomerative clustering (Mitchell, 1997). The algorithm creates a hierarchy of groups based on their spatial arrangement (see Section 5.3 of this thesis). The resulting object groups are represented in the same way as any other object in the environment: as a list of features. Most of the features representing an object group are the same as those for single objects (e.g., location, dimensions, color, ...). This similar representation makes it possible to reuse the concepts that are used to describe single objects (e.g., *big*, *far*, etc.) for describing object groups. However, there are some features that are only present in object groups (such as *density* and *number of constituents*). This makes it possible to have concepts that can only modify object groups (e.g., *two* or *many*).

In sum, the present model permits the agents to use NP's and CN's to describe object sets. The NP's have a determiner position that is always filled by a quantifier. The difference between quantifiers and adjectives has both a syntactic and a semantic side. Semantically, quantifiers are limited to the application to object groups, whereas adjectives can modify both groups and single objects. Syntactically, quantifiers can only occur at the head of the utterance (due to the NP-rule), whereas adjectives can be at any position before the noun (due to the CN-rule).

## 8.3 Baseline Experiment

As mentioned above, in previous experiments it was shown that agents are likely to derive gradable quantifiers from adjectives (Pauw, 2013b). In these experiments agents did indeed develop gradable quantifiers that are similar to the English *many* and *few*. However they emerge as adjectives, not as quantifiers. In this chapter we look at how they grammaticalize into quantifiers. To this end it is no longer assumed that the grammatical categories are fixed, but that the agents have some mechanism that reanalyzes the grammatical category of lexical entities called a reanalysis operator.

Before describing the precise mechanics of this operator, I'll take a small detour and first speculate on its possible consequences. This section discusses a baseline experiment in which I compare the performance of the *initial language* (where *many* and *few* are adjectives) to the hypothetical *target language* (where *many* and *few* are quantifiers).

The initial language is a copy of the language that came out of the experiment described in Pauw (2013b). For this language, the lexicon consists of the adjectives *left*, *right*, *front*, *back*, *far*, *near*, *big*, *small*, *few*, and *many*; the nouns *block*, *box* and *robot*; and the numeral quantifiers *one* to *twelve*. Additionally, the lexicon contains the CN and NP rule as described in the previous section. The target language is identical, but the gradable quantifiers *few* and *many* are defined as quantifiers. As a consequence, in the target language the gradable quantifiers are not parsed using the CN rule, but rather using the NP rule. Syntactically this means that *many* and *few* can only occur at the head of the noun phrase, whereas in the initial language they can occur on any adjective position. Semantically, this means that the application of these gradable quantifiers is restricted to only groups (sets) of objects.

### 8.3.1 Baseline Communicative Success

We can now compare the performance of these two languages in a language game experiment. This experiment follows the script as described in Section 8.2, but the agents do not do any reanalysis yet. I let the agents play 5000 interactions and monitor the average communicative success. The experiment was repeated for both languages (the initial language and the target language). The graph in Figure 8.2 shows the results. The graph shows three different results: 1) the average communicative success of all the language games where the speaker picked a single object as referent, 2) the average communicative of the language games where speaker picked a group of objects, and 3) the two combined (the average communicative success of all the language games).

We can see that there is no difference in communicative success between the two languages for any of those conditions. For condition 2 this result is hardly surprising: The only semantic difference between the two languages is that in the

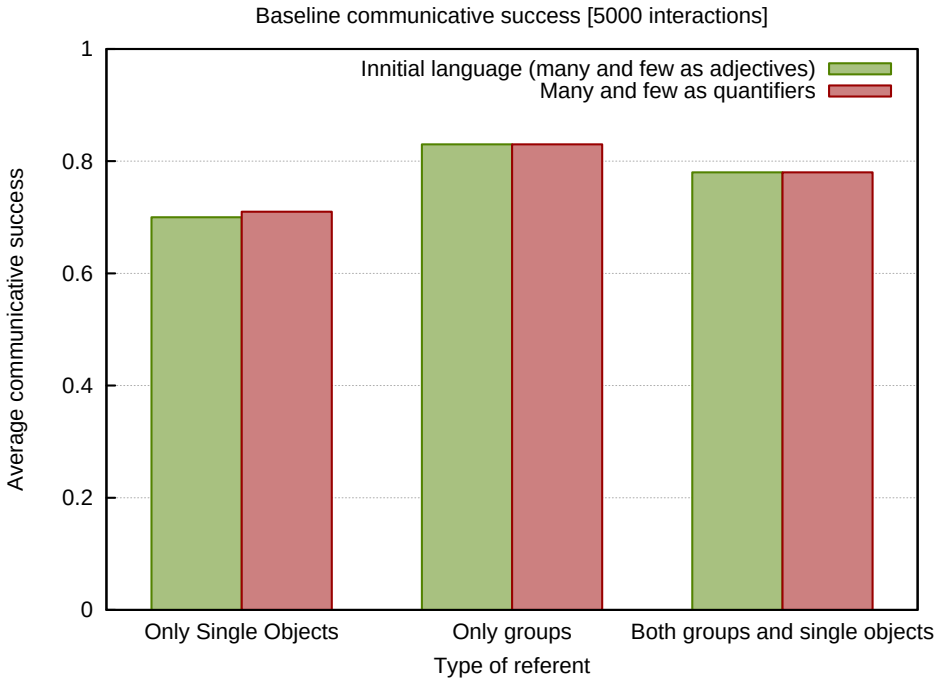



Figure 8.2: Baseline communicative success for the initial language (*many* and *few* as adjectives) and the target language (*many* and *few* as quantifiers). The bars represent the average communicative success taken over 5000 interactions. The results shows three different conditions for the referent type: 1) the success of interactions where the referent that is described by the speaker is a single objects; 2) the success of interactions where the speaker describes only groups of objects; and 3) both combined. There is no difference in communicative success for any of the conditions between the initial and the target language.

target language *many* and *few* are quantifies, which means that they can only apply to groups of objects. For condition 2 the referent is always a group of objects anyway, so for this condition there can not possibly be any difference.

But, it is perhaps less evident that there is no difference for single objects either. This is due to the way these gradable quantifiers are used: Gradable quantifiers can not make meaningful distinctions in single objects. It is, for example, meaningless to say something like "a many block," even if it would be grammatically correct. In other words, pragmatically, the gradable quantifiers can only be applied to groups of objects not to single objects, even if this is not explicitly marked by the grammar. In the present model this intuition is captured by the fact that gradable quantifiers, when applied to a single object, will always return a score of 0. This means that gradable quantifiers do not create any contrast between single objects and are therefore not helpful in describing them. So, for the communicative success it does not matter if the grammar considers gradable quantifiers to be adjectives or quantifiers.

In other words, in terms of communicative success there is no difference in performance between the two language. This means that communicative success is not likely to be an incentive for the proposed category shift. So why would agents reanalyze the grammatical category? If communicative success is not an incentive, what is?

### 8.3.2 Baseline Cognitive Effort

The main difference between the grammatical category of quantifiers and adjectives in the present grammar (syntactic patterns aside), is the fact that for quantifiers the application is restricted to sets of objects. In the previous section we saw that for the gradable quantifiers *few* and *many* this difference does not affect the communicative success.

It does, however, affect the search space. Before starting interpretation, any entity in the environment (single objects and groups of objects alike) is a candidate referent. The role of the defined language is not only to compute the referent, but also to do so efficiently. As soon as a quantifier is encountered, the semantic system knows it can ignore single objects and reduce the evaluation to groups of objects. Which reduces the search space. The effect is especially strong for the present language because quantifiers are the head of the noun phrase, making it the first item that is evaluated. Thus reducing the search tree right from the start.

So, in order to understand the effects of the quantifier category, we have to consider this processing efficiency. To this end, I adopt a common approach in AI to use the size of the search tree as a measure for *cognitive effort* (Steels and Wellens, 2006) (also sometimes referred to as *processing effort* (van Trijp, 2012)). The search tree size is linearly proportional to the amount of objects that have to be considered for evaluation. The precise values are not relevant. In order to make

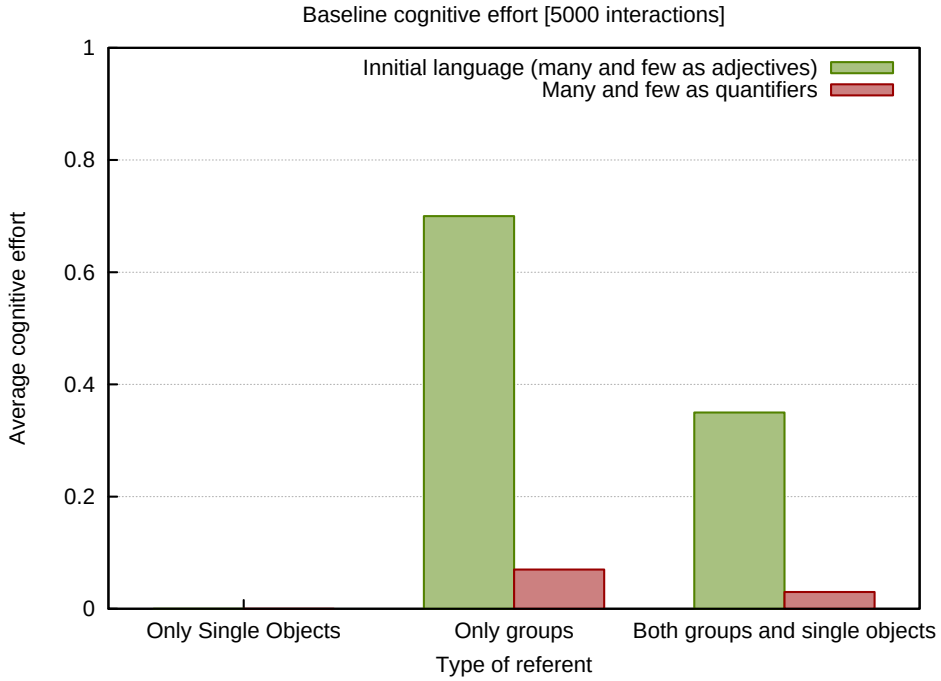



Figure 8.3: Baseline cognitive effort for the initial language (*many* and *few* as adjectives) and the target language (*many* and *few* as quantifiers). The bars represent the average cognitive effort taken over 5000 interactions. The values are normalized between 0 and 1. The results show that for single objects, there is no difference in cognitive effort. However, when the speaker describes a group of object, for the target language (which treats gradable quantifiers as quantifiers), the cognitive effort is much lower.

the result comparable across language games, the cognitive effort is normalized to yield a score between 1 and 0, where 1 is the maximum amount of objects that the agent could consider in a specific scene and 0 the minimum. In order to avoid any confusion that may arise, it should be emphasized that cognitive effort in this context is defined as a function of the semantical search space. The search that is required for parsing the sentence, is not taken into consideration.

Figure 8.3 shows the results of the same experiment as above, but showing the cognitive effort instead of the communicative success. We can see a dramatic difference in cognitive effort for object groups (the second condition). The amount of search that the agents have to do for describing object groups is much lower

when gradable quantifiers are modeled as quantifiers. And, although there is no difference in communicative success between the initial and target language, we see that, overall, the cognitive effort is much lower for the target language (where gradable quantifiers are quantifiers). So, unlike communicative success, reducing cognitive effort might be a good incentive for the agents to grammaticalize gradable quantifiers into quantifiers.

### 8.3.3 Baseline Lateral Stability

The previous two sections established the particular communicative pressure that might trigger the reanalysis operator to convert *many* and *few* into quantifiers. Yet, before going into the details of a reanalysis operator, there is another question we have to tackle: Why does a mechanism that changes the grammatical categories of *many* and *few* leave all other lexical items alone? For example, why aren't the categories of *big* and *small* not altered in the same way as *many* and *few*? Or why doesn't the reanalysis mechanism change *left* into a noun, or *three* into an adjective? If we want make any claims of generality at all we can not simply assume that somehow the reanalysis operator is limited to affect only gradable quantifiers.

Consider, for example, the adjectives *big* and *small*. What would happen if they are defined as quantifiers? Remember that any concept that can be used to describe single objects can also be used to describe object groups. So it is perfectly consistent to define *big* and *small* as quantifiers, they are already used by the agents to describe object groups. The difference resulting from defining them as adjectives is that their application is restricted to only object groups. To see the consequences, I have implemented a hypothetical (and, in this case, undesirable) language in which *big* and *small* are defined as quantifiers, and compared it to the initial language. Figure 8.4 shows the result for cognitive effort, and Figure 8.5 shows the results for communicative success.

We can see just as for the gradable quantifiers, that the cognitive effort is lower when *big* and *small* are defined as quantifiers. The effect is much smaller than for the gradable quantifiers, but it's there. However, if we now look at Figure 8.5, we see that this minor decrease in cognitive effort comes at a very high cost: a very strong decrease in communicative success. The reason for this decrease is that if *big* and *small* are defined as quantifiers, their application is restricted to sets of objects. For gradable quantifiers this was not a problem, since they do not make meaningful distinctions between single objects anyway. But, for *big* and *small* this does not hold; they do make meaningful distinctions for single objects, and therefore restricting them to sets seriously hampers the language. So, if the reanalysis mechanism is to take both cognitive effort and communicative success into account, it is not likely to affect *big* and *small*.

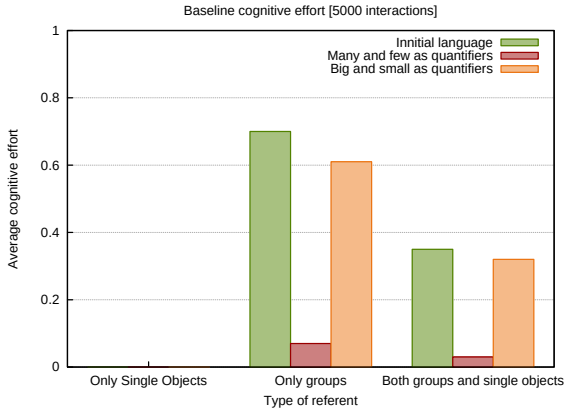



Figure 8.4: Baseline cognitive effort for the undesirable language (where *big* and *small* are treated as quantifiers), the initial language (*many* and *few* as adjectives) and the target language (*many* and *few* as quantifiers). The bars represent the average cognitive effort taken over 5000 interactions. The results show that there is a slightly lower cognitive effort for the undesirable language. Although the effect is not nearly as dramatic as with the target language.

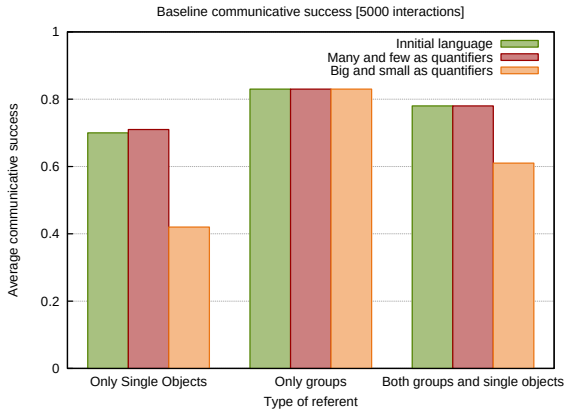



Figure 8.5: Baseline communicative success for the undesirable language (where *big* and *small* are treated as quantifiers), the initial language (*many* and *few* as adjectives) and the target language (*many* and *few* as quantifiers). The bars represent the average communicative success taken over 5000 interactions. The results show a much lower communicative success for the undesirable language than for the initial and target language.

## 8.4 Selection Experiment

The previous section showed that a combination of communicative success and cognitive effort might provide the right selective pressures for the agents to re-analyze *many* and *few* as quantifiers, but leave *big* and *small* unbothered as adjectives. To this end I manually created two languages to compare with the initial language. There are, however, many more linguistic variations imaginable that could be tested in the same way. In fact, there are way too many possible languages to test all of them individually (with the 10 modifiers and 2 categories of the example language we can already define $2^{10} = 1024$ different languages). So instead, I let the agents select the optimal languages themselves in a *selection experiment*. In the selection experiment the agents use a very general selectionist mechanism to find the optimal language given a set of constraints. In this case, linguistic variation is created by providing the agents with both an adjective and a quantifier versions of every modifier (any lexical item that is not a noun). So, every modifier has two competing versions. The task for the agents is for every modifier to select the fitter version of the two.

This is done by assigning a *confidence score* (or *construction score*) (between 0 and 1) to every lexical item, that reflects its performance. After every successful interaction the hearer will increase the confidence score of every lexical item that was involved in that particular interaction. If the interaction was not successful the confidence score of all the lexical items involved is decreased. The scores are updated using the following *update function*:

$$S_{n+1}(l) = S_n(l) + \lambda S_n(l)(1 - S_n(l))$$

Where $S_n(l)$ is the original score, $S_{n+1}(l)$ is the updated score of lexical item $l$, and $\lambda$ is a rate at which a single update influences the score. For successful interactions $\lambda = 0.1$ for unsuccessful interactions $\lambda = -0.1$. The function is a numerical approximation of an S-curve. This means that when the score gets closer to its extreme values (0 or 1), the rate at which the score changes gets lower. Update functions with this property have proven to be more stable than linearly updating scores for similar experiments; i.e., a false positive or negative has less influence on the confidence score.

Since there are multiple versions of every lexical item in the lexicon of the agent, they need some way of deciding which one they use. This is done by picking one randomly, using the scores of the lexical items as a distribution. So, lexical items with a high score have a bigger chance of being used than the ones with a low score.

Using the given initial variation and the update function, I let the agents play a series of language games. Figure 8.6 shows the results. It shows the average scores of different groups of lexical items. (Lexical items are grouped together for the sake of readability.) The graph clearly shows that for the lexical items, *big*, *small*, *left* and *right*, the agents converge on their adjectival variant. This

is not surprising: The quantificational variants are much less successful. They are limited to describe object groups whereas the adjectival variants can be used both to describe single objects and groups of objects, so the adjectival variants are being used successfully much more often.

For the words *many* and *few*, on the other hand, Figure 8.6 does not show any convergence on either their adjectival or quantificational variant. This is due to the problem discussed in the previous section: The gradable quantifiers *many* and *few* do not make meaningful distinctions for single objects. Therefore, the grammatical category they belong to does not influence the communicative success. And, since the update function is only affected by communicative success, neither variant will prevail over the other.

Figure 8.7 shows what happens if we also take cognitive effort into account. The $\lambda$ in the update function is not statically defined as above, but takes the cognitive effort ($C$) into account by letting it influence the scoring rate ($\lambda = 0.2 \cdot (1 - C)$ for success and $\lambda = -0.2 \cdot C$ for failure[1]). Now we can see that for the words *many* and *few*, the agents converge on their quantificational variant in about 80% of the time.

In sum, the grammatical categories of the lexical items were not fixed in the initial language, but the agents were provided with all options. Using a basic selectionist mechanism, the community automatically converged on a fixed categorization of all the lexical items. The modifiers in the lexicon that predicate both single objects and object groups (e.g., *big*, *small*, *left* and *right*) are categorized as adjectives, and the modifiers that only predicate object groups (e.g., *many* and *few*) are more likely to be categorized as quantifiers. Where the latter only happens if the cognitive effort is taken into account.

## 8.5 Grammaticalization Experiment

The previous section illustrated the the working of the selectionist mechanism that allows the agents to converge on an optimal categorization provided a given variation. This section describes an experiment where agents have to create this variation themselves.

The agents will start out with the same initial language as described in Section 8.3 (i.e., *few*, and *many* are defined as adjectives). But, now we provide the agents with a *reanalysis operator*: a mechanism that allows them to reanalyze the grammatical categories of the lexical items in their lexicon. The reanalysis operator is used when an interaction is not successful or the cognitive effort is too high. It introduces a copy of the item under analysis but with a different grammatical category. So after applying the reanalysis operator, there are two competing versions of the same lexical item.

[1]Note that the cognitive effort is normalized to average out on $C = 0.5$, so on average $\lambda = 0.1$ — the same as for the previous graph.

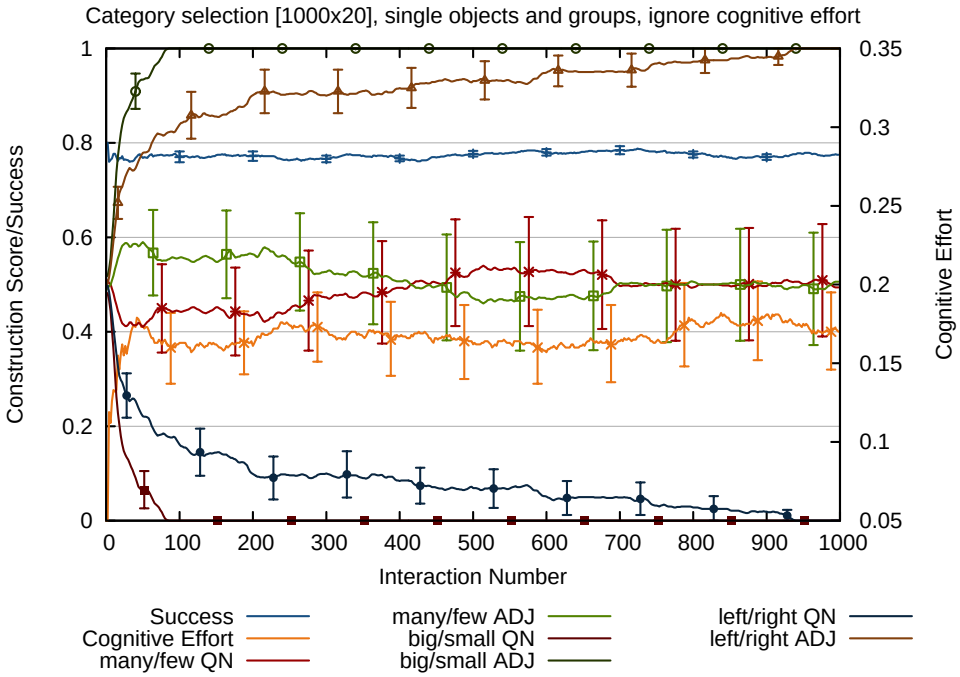



Figure 8.6: The selection of lexical items. Thes graphs follow the scores of specific lexical items over 1000 interactions. Also, the communicative success and the cognitive effort are shown. The experiment is repeated 20 times, the results are averaged. Every modifier has two variants: one adjectival (ADJ), and one quantificational (QN). They all start with a score of 0.5. The selection mechanism only regards the communicative success. After few hundred interactions it becomes clear that for the modifiers *left*, *right*, *big*, and *small*, the adjectival variants are selected. However, the community of agents does not converge on either the QN or ADJ variant of *many* and *few*. The cognitive effort and communicative success are stable over the course of the experiment.

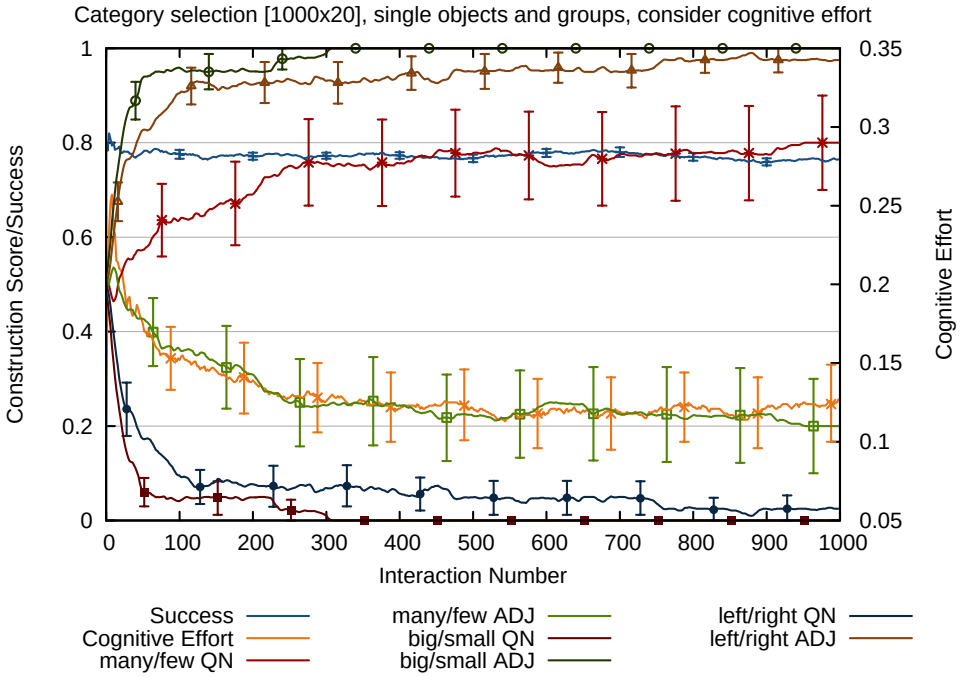



Figure 8.7: The selection of lexical items while minimizing cognitive effort. These graphs follow the scores of specific lexical items over 1000 interactions. Also, the communicative success and the cognitive effort are shown. The experiment is repeated 20 times, the results are averaged. Every modifier has two variants, one adjectival (ADJ), and one quantificational (QN). They all start with a score of 0.5. The selection mechanism regards both the communicative success and the cognitive effort. After a few hundred interactions it becomes clear that for the modifiers *left*, *right*, *big*, and *small*, the ADJ variants are selected. For the modifiers *many* and *few* the agents converge on the QN variant. As a result, the cognitive effort decreases over the course of the experiment. The communicative success remains the same.

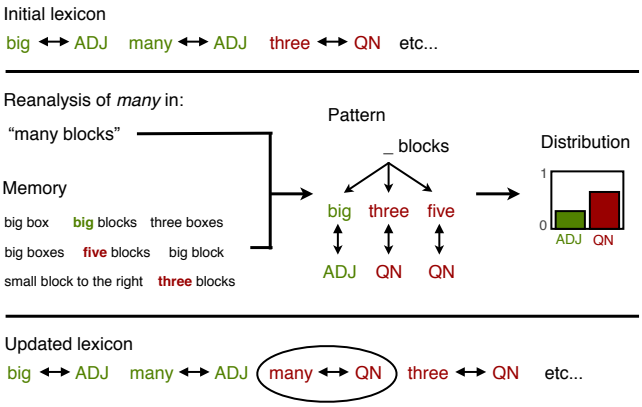



Figure 8.8: The reanalysis of *many* in the utterance "many blocks". In the initial lexicon *many* is an adjective. The operator looks up what other lexical items occurred at the same slot. In this case, those are *big*, *three* and *five*, thus one adjective and two quantifiers. This gives a distribution of grammatical categories. A new grammatical category is picked from that distribution (QN in this example). Using this category, a new lexical item (*many* as quantifier) is created and added to the lexicon.

### 8.5.1 Reanalysis Operator

The reanalysis operator is based on the notion of an *attractor pole* as presented in Bisang (1998) and Sommerer (2012). The general idea is that if a slot in an utterance correlates with a specific grammatical category, it encourages lexical items that occur at that slot to grammaticalize into it. For example, in our grammar, quantifiers occur at the head of an utterance. If an adjective like *many* occurs in the same position, it might be invited to grammaticalize into a quantifier as well.

A lexical item is reanalyzed with respect to a problematic utterance (i.e., an utterance that was used in an unsuccessful interaction or that required a high cognitive effort in parsing). Figure 8.8 provides an outline of the working of the reanalysis operator for the lexical item *many* in the utterance "many blocks". The operator requires, first of all, a memory of earlier conversation. To keep things simple, the agents keep track of every utterance they ever used. Now, to reanalyze *many* in "many blocks", first the pattern "$X$ blocks" (where $X$ can be any single lexical item) is looked up in the memory of the agent. This way, the agents establish a list of co-occurring lexical items. In the present example the agents heard the utterances "big blocks", "three blocks" and "five blocks" before. So, the lexical items *big*, *three*, and *five* co-occur with *many* for the slot $X$ in "$X$ blocks".

Second, a distribution over grammatical categories for that slot is established:

All these lexical items have a specific grammatical category. By simply counting these grammatical categories the agent can establish a distribution over grammatical categories for the slot. In this case, slot $X$ has a $[\frac{1}{3}, \frac{2}{3}]$-distribution over adjectives and quantifiers respectively (since *big* is an adjective, and *three* and *five* are quantifiers).

Using this distribution the agents pick a new category for the lexical item. If this is a new category for that lexical item, it is added to the lexicon of the agent. For the present example, *many* has a two out of three chance of being reanalyzed as a quantifier. If this happens, the agent adds the new version of the lexical item to the lexicon with an initial confidence score of 0.5. So, after application of the reanalysis operator, the lexicon contains two versions of the lexical item *many*, one as an adjective, and one as a quantifier. At this stage, both versions of the lexical item are available for parsing and production.

It should be noted about this mechanism that it is extremely general. Nothing is put into it to make it focus specifically on turning adjectives into quantifiers. It simply reanalyzes based on emerging surface patterns. It can assign any grammatical category to any lexical item if the produced utterances would give rise to it. In fact, the reanalysis operator is so general that is completely agnostic to the grammar that is being used. If we would define a different grammar, with, for example, more grammatical categories (e.g., verbs or prepositions), these categories would be automatically become available to the reanalysis operator. Deciding on which reanalyses are useful and which are not is not up to the present operator, but to the selectionist mechanism as described in the previous two sections.

### 8.5.2 Experiment

Now we can put everything together. In Section 8.4 it was shown how a community of agents can use a selectionist framework to find an optimal language given a specific set of linguistic variations. This framework encourages variants of lexical items that yield a high communicative success and require a low cognitive effort. In the previous section I illustrated the workings of the reanalysis operator that creates this variation. What happens when we put the selection mechanism and the reanalysis operator together?

In this last experiment I provide a community of agents with the initial language as described in Section 8.3 (i.e., *many* and *few* are defined as adjectives). Using this language I let the agents play a series of language games. The language games are the same as described in the previous sections, but now we provide the agents with a reanalysis operator. Every time, a language game is not successful or the parsing of the involved utterance requires a high cognitive effort, the reanalysis operator is called.

The learning parameters are identical to the ones described in Section 8.4: After every successful interaction, the hearer increases the scores of the involved lexical items. If the interaction was not successful the scores are decreased. The

rate at which the scores increase or decrease is determined by the cognitive effort. This experiment is repeated 10 times for 4000 interactions.

Figure 8.9 and 8.10 show the results. They show the cognitive effort, the communicative success, and the scores of the lexical items over time. The figures show the same results, but Figure 8.9 shows individual lexical items and Figure 8.9 shows the lexical items grouped together in semantic groups (*many* and *few*—gradable quantifiers; *big* and *small*—adjectives; *front*, *back*, *left* and *right*—adjectives).

The scores of the lexical item at the first interaction correspond to the initial language: *many* and *few* start out as adjectives. The first thing to note about the graph in Figure 8.9 is that the scores of *many* and *few* for their adjectival variants systematically decrease over time, while the scores of their newly created quantificational counterparts increase at the same time. The figure shows that the quantificational variant of *few* overtakes the adjectival one at around interaction 450, and for *many* the same happens between interaction 500 and 900. After 3000 interactions we see the scores stabilize. The lexical items *many* and *few* are now quantifiers.

Furthermore, the graph shows a lot of variation being created. Not only *many* and *few* are being reanalyzed, but all lexical items are at some point in the experiment. These other variations, however, do not persist. The graph in Figure 8.10 shows this more clearly. The quantificational variants of other adjectives are being tentatively created by the agents, but their scores stay low. Thus, adjectives other than *many* and *few* stay adjectives.

The graphs further show, that the grammaticalization of the gradable quantifiers into quantifiers has no effect on the communicative success. The communicative success stays stable around 80% over the 4000 interactions. But, it does have a very strong effect on the cognitive effort. Reanalyzing *many* and *few* reduces the search space dramatically.

These results are not surprising; they essentially present a corollary of the results presented in Section 8.3 and 8.4. Since gradable quantifiers do not make meaningful distinctions for single objects, they can be grammatically confined to sets of objects. Doing so reduces the cognitive effort. So the quantifier versions of these gradable quantifiers are preferred over their adjectival ones. The lateral stability (the fact that other adjectives do not grammaticalize into quantifiers) is due to the fact that these other adjectives do make meaningful distinctions for single objects, and therefore, confining them to sets by turning them into quantifiers would make the language less successful.

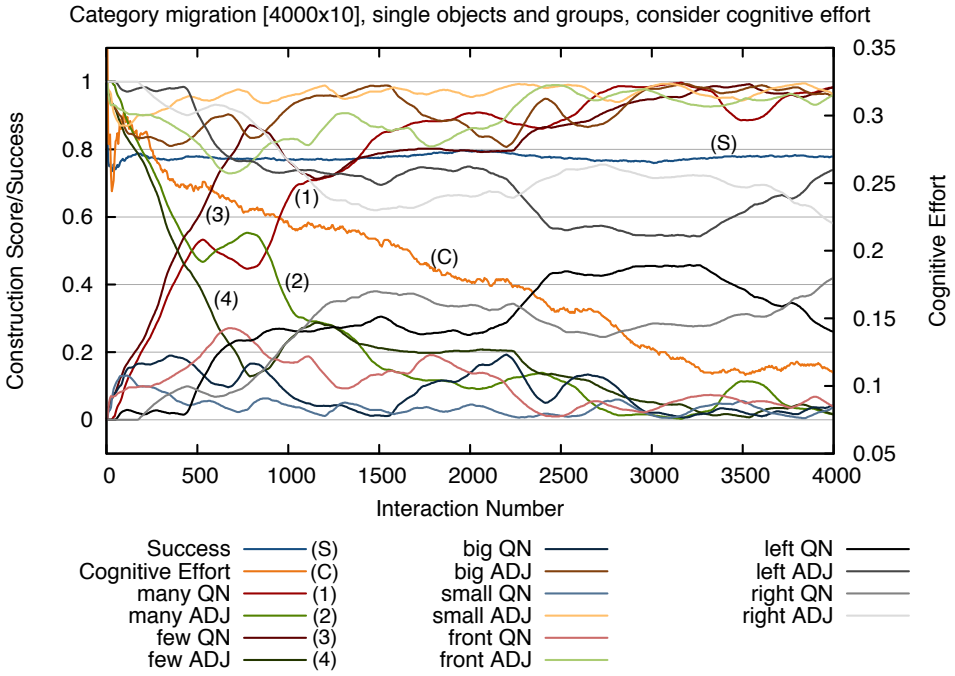



Figure 8.9: The grammaticalization of quantifiers. This graph shows the progression of scores of the lexical items in the lexicon of the one agent over 4000 interactions. The experiment is repeated 10 times. The results are averaged. Figure 8.10 shows a more comprehensible versions of this graph. The agents starts out with the initial language described in 8.3 (where the modifiers *left*, *right*, *big*, *small*, *many*, and *few* are all adjectives). The graph shows that the reanalysis mechanism introduces quantificational variants of all these modifiers at some point. However, most of them are not successful. Only the quantificational versions of *many* and *few* increase in score and overtake their quantificational variants between interaction 400 and 1000. Their adjectival variants gradually disappear from the lexicon of the agent. As a result, the cognitive effort shows a steady decrease, while the communicative success stays the same.

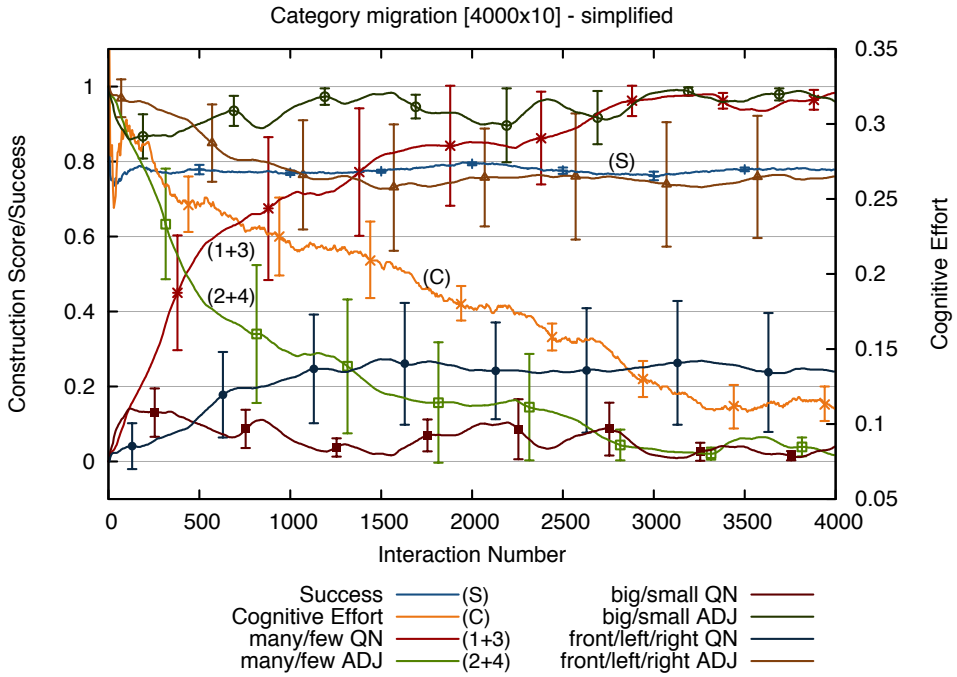



Figure 8.10: The grammaticalization of quantifiers. This graph shows the progression of the average score of the gradable quantifiers and the average score of other modifiers over 4000 interactions. The experiment is repeated 10 times. The results are averaged. This is graph is a more comprehensive version of the graph in Figure 8.9. The reanalysis mechanism introduces quantificational variants of all modifiers. But, only the quantificational variants of gradable quantifiers become successful. As a result, the cognitive effort shows a steady decrease, while the communicative success stays the same.

## 8.6 Discussion

### 8.6.1 Conclusion

The experimental results show how the development of the gradable quantifiers *many* and *few* in a community of robotic agents follow the speculated grammaticalization path: the grammatical category of gradable quantifiers moves from adjective to quantifier due to the cognitive overlap between quantifiers and gradable quantifiers. The resulting language has shown to be more efficient in processing than the initial language while preserving the communicative success. It is this increased efficiency (or lower cognitive effort) that provides the incentive for the gradable quantifiers to become quantifiers.

Other adjectives (such as *big* and *small*) on the other hand, do not grammaticalize into quantifiers. In principle the reanalysis mechanism could also turn these adjectives into quantifiers, and this would also reduce the cognitive effort. But, this would be at the cost of a much lower communicative success, which remains the most important selection criterion. For the same reason other category shifts (e.g., common noun to adjective or quantifier to common noun) are not observed in this experiment.

These results do not stand on their own. In the previous chapter it was shown how gradable quantifiers are likely to emerge as adjectives in a community of agents due to the cognitive relation between number and size. Overall, together with the present chapter, these results show how gradable quantifiers can evolve in a language. They also highlight how cognitive overlap between number motivates their dual nature; i.e., the fact that *many* and *few* can act like both quantifiers and adjectives.

On a more general note, this research is part of a recent effort that studies known grammaticalization patterns, and uses situated interaction games to find a cognitive motivation for those patterns (van Trijp, 2012; Beuls and Steels, 2013). These studies, including the present experiment, suggest how the inclusion of cognitive constraints may illuminate some of the grammaticalization patterns found in human languages.

One can wonder how general these results are. Don't they depend too much on model-specific assumptions? Of course, there are a number implicit and explicit assumptions in the model. For example, the use of a prototype theory to model the semantics, the pre-established joint attention, the fact that the goal of the game is shared knowledge, and the assumption that categories are not fixed but that they can change based on the surface patterns they occur in. Some of these assumptions play an important role in the outcome of this experiment (e.g, the working of the reanalysis mechanism). And it is of course impossible to defend that the model mimics human cognition perfectly.

However, the model is inspired by cognitive studies. The representation of number approximation is at least reminiscent of the human representation in

that it relies on features of density and size. Secondly, by keeping the model as simple and transparent as possible and breaking down the experiment into the steps that are described in this chapter, I have shown how the individual assumptions influence the outcome of the experiments. Furthermore, it should be emphasized that the crucial parts of the experiment are very general, non domain-specific, mechanisms. For example, the selection mechanism is a general mechanism that has been applied to many evolutionary experiments in different linguistic domains. Also, the reanalysis mechanism was not biased to target gradable quantifiers. It can change the grammatical category of any lexical item into any other category. So, nothing in the reanalysis mechanism or the selectionist model is tailored towards the specific question of gradable quantifiers. It is important to realize that therefore, the fact that only gradable quantifiers are affected and no other lexical items, is not a particularity of the present model. It can only be attributed to the linguistic maxims; i.e., *a priori* any lexical item could get any grammatical category, but only the grammaticalization of gradable quantifiers provides a communicative advantage.

### 8.6.2 Future research

One essential assumption was not addressed in this article: the existence of a quantifier category. The reanalysis mechanism relies on existing grammatical categories. If the agents did not have any other quantifier in their lexicon, they would never grammaticalize gradable quantifiers into it. To my opinion, the most interesting topic for further research is the emergence of the grammatical category of quantifiers itself. So, how does a language go from having no quantifiers at all, to at least one quantifier? Which hopefully would shed some light on the more general question: Where do grammatical categories come from?

The reanalysis mechanism relies on another assumption that is open to future research. The reanalysis is, as mentioned before, agnostic to the particular grammar used. But, for the present experiment it is assumed that looking at surface patterns is enough. For the relatively simple grammar as presented in Section 8.2 this works fine. But, for more elaborate grammars, a deeper structural analysis might be required. It would be interesting to see if existing formalisms (such as DOP (Bod et al., 2003)) could be used to create a more general reanalysis mechanism.

Studying more complex grammars would also open the door to other interesting aspects of quantification. This research focuses on noun phrases that describe either objects or sets of objects. But, in more complex sentences, the role of quantifiers becomes more than just these descriptions. For example, quantifiers play an important role in inference. These uses of quantifiers have been extensively studied in the field of formal semantics. Another interesting topic of further investigation would be to see how this well documented inferential use of quantifiers emerges in a language.

# Part IV

# End Matter

# Bibliography


Allik, J. and Tuulmets, T. (1991). Occupancy model of perceived numersoity. *Perception & Psychophysics*, 49:303–314.

Antell, S. and Keating, D. (1983). Perception of numerical invariance in neonates. *Child Development*, 54(3):695–701.

Barlow, H. (1978). The efficiency of detecting changes of density in random dot patterns. *Vision research*, 18(6):637–650.

Barwise, J. and Cooper, R. (1981). Generalized quantifiers and natural language. *Linguistics and philosophy*, 4(2):159–219.

Belpaeme, T. (2002). *Factors influencing the origins of colour categories*. PhD thesis, Vrije Universiteit Brussels (VUB), Brussels, Belgium.

Bergen, B. and Chang, N. (2005). Embodied construction grammar in simulation-based language understanding. *Construction grammars: Cognitive grounding and theoretical extensions*, pages 147–190.

Beuls, K. and Steels, L. (2013). Agent-based models of strategies for the emergence and evolution of grammatical agreement. *PLoS One*. Accepted.

Bevan, W., Maier, R. A., and Helson, H. (1963). The influence of context upon the estimation of number. *The American Journal of Psychology*, 76(3):464–469.

Bevan, W. and Turner, E. (1964). Assimilation and contrast in the estimation of number. *Journal of Experimental Psychology*, 67(5):458.

Bisang, W. (1998). Grammaticalization and language contact constructions and positions. *Typological Studies In Language*, 37:13–58.

Blakemore, D. (1992). *Understanding utterances*. Blackwell Publishing.

Blakemore, D. (2002). *Relevance and linguistic meaning: The semantics and pragmatics of discourse markers*. Cambridge University Press.

Bleys, J. (2008). Expressing second order semantics and the emergence of recursion. In D.M. Smith, A., Smith, K., and Ferrer i Cancho, R., editors, *Proceedings of the 7th International Conference on the Evolution of Language (EVOLANG 7)*, pages 34–41, Singapore. World Scientific Publishing.

Bleys, J., Loetzsch, M., Spranger, M., and Steels, L. (2009). The grounded colour naming game. *Proceedings Roman-09*.

Bod, R., Scha, R., and Sima'an, K. (2003). *Data-oriented parsing*. CSLI Publications.

Borning, A. (1981). The programming language aspects of thinglab, a constraint-oriented simulation laboratory. *ACM Trans. Program. Lang. Syst.*, 3(4):353–387.

Chomsky, N. (1986). *Knowledge of language: Its nature, origin, and use*. Praeger Publishers.

Clearfield, M. and Mix, K. (2001). Amount versus number: Infants' use of area and contour length to discriminate small sets. *Journal of Cognition and Development*, 2(3):243–260.

Cooper, R. (1984). Early number development: Discovering number space with addition and subtraction. *Origins of cognitive skills*, pages 157–192.

Coventry, K., Cangelosi, A., Newstead, S., Bacon, A., and Rajapakse, R. (2005). Grounding natural language quantifiers in visual attention. *P Cogn Sci Soc*, pages 506–511.

Coventry, K., Cangelosi, A., Newstead, S., and Bugmann, D. (2010). Talking about quantities in space: Vague quantifiers, context and similarity. *Language and Cognition*, 2(2):221–241.

Durgin, F. (1995). Texture density adaptation and the perceived numerosity and distribution of texture. *Journal of Experimental Psychology: Human Perception and Performance*, 21(1):149.

Durgin, F. (2008). Texture density adaptation and visual number revisited. *Current Biology*, 18(18):R855–R856.

Eschenbach, C. and Kulik, L. (1997). An axiomatic approach to the spatial relations underlying left-right and in front of-behind. In *KI-97: Advances in Artificial Intelligence*, pages 207–218. Springer.

Everett, D. (2005). Cultural constraints on grammar and cognition in pirahã. *Current Anthropology*, 46(4):621–646.

Feigenson, L., Carey, S., and Spelke, E. (2002). Infants' Discrimination of Number vs. Continuous Extent* 1. *Cognitive Psychology*, 44(1):33–66.

Fernando, T. and Kamp, H. (1996). Expecting many. In *Proceedings of SALT*, volume 6, pages 53–68.

Fodor, J. (1978). Tom Swift and his procedural grandmother. *Cognition*, 6(3):229–247.

Fodor, J. (1979). In Reply to Philip Johnson-Laird. *Cognition*, 7(1):93–95.

Fodor, J. A. (1983). *The Modularity of Mind: An Essay on Faculty Psycholgy*. The MIT Press.

Foley, J. M. (1980). Binocular distance perception. *Psychological Review*, 87(5):411–434.

Freksa, C. (1999). Links vor – Prototyp oder Gebiet? In Rickheit, G., editor, *Richtungen im Raum*, pages 231–246. Westdeutscher Verlag.

Fujita, M., Kuroki, Y., and Ishida, T. (2003). A small humanoid robot sdr-4x for entertainment applications. In *Advanced Intelligent Mechatronics, 2003. AIM 2003. Proceedings. 2003 IEEE/ASME International Conference on*, volume 2, pages 938–943.

Fujita, M., Sabe, K., Kuroki, Y., Ishida, T., and Doi, T. T. (2005). Sdr-4x ii: A small humanoid as an entertainer in home environment. In Dario, P. and Chatila, R., editors, *Robotics Research*, Springer Tracts in Advanced Robotics, pages 355–364. Springer.

Gärdenfors, P. (2004). *Conceptual spaces: The geometry of thought*. The MIT Press.

Gerasymova, K. and Spranger, M. (2012a). An Experiment in Temporal Language Learning. In Steels, L. and Hild, M., editors, *Language Grounding in Robots*. Springer, New York.

Gerasymova, K. and Spranger, M. (2012b). Handling temporal language: a case study for Russian aspect. In Steels, L. and Hild, M., editors, *A Whole Systems Approach to Grounding Language in Robots*. Springer.

Ginsburg, N. (1978). Perceived numerosity, item arrangement, and expectancy. *The American journal of psychology*, pages 267–273.

Glöckner, I. (2006). *Fuzzy quantifiers: a computational theory*. Springer Verlag.

Goldberg, A. (1995). *Constructions: A construction grammar approach to argument structure*. University of Chicago Press.

Goocher, B. E. (1965). Effects of attitude and experience on the selection of frequency adverbs. *Journal of Verbal Learning and Verbal Behavior*, 4(3):193–195.

Halberda, J. and Feigenson, L. (2008). Developmental change in the acuity of the number sense: The approximate number system in 3-, 4-, 5-, and 6-year-olds and adults. *Developmental Psychology*, 44(5):1457–1465.

Healey, G. and Kondepudy, R. (1994). Radiometric CCD Camera Calibration and Noise Estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16:267–276.

Hermann, T. and Grabowski, J. (1976). *Psychologie der Objektbenennung*. Hans Huber Verlag.

Herskovits, A. (1986). *Language and spatial cognition*. Studies in Natural Language Processing. Cambridge University Press.

Hewitt, C. (1969). Planner: A language for proving theorems in robots. *Proc First Int'l Joint Conf. in Artificial Intelligence*.

Hild, M., Siedel, T., Benckendorff, C., Thiele, C., and Spranger, M. (2012). Myon, a New Humanoid. In Steels, L. and Hild, M., editors, *Language Grounding in Robots*. Springer, New York.

Hormann, H. (1983). The calculating listener or how many are einige, mehrere, and ein paar (some, several, and a few). *Meaning, use, and interpretation of language*, pages 221–234.

Jevons, W. (1871). The power of numerical discrimination. *Nature*, 3:281–282.

Johnson-Laird, P. N. (1977). Procedural semantics. *Cognition*, 5(3):189–214.

Johnson-Laird, P. N. (1978). What's wrong with Grandma's guide to procedural semantics: A reply to Jerry Fodor. *Cognition*, 6(3).

Kamp, H. (1975). Two theories about adjectives. *Formal semantics of natural language*, pages 123–155.

Kamp, H. (1981a). The paradox of the heap. *Aspects of philosophical logic*, pages 225–277.

Kamp, H. (1981b). A theory of truth and semantic representation. *Formal Semantics*, pages 189–222.

Kaufman, E., Lord, M., Reese, T., and Volkmann, J. (1949). The discrimination of visual number. *The American journal of psychology*, 62(4):498–525.

Kelleher, J. and Costello, F. (2005). Cognitive representations of projective prepositions. In *Proceedings of the Second ACL-Sigsem Workshop of The Linguistic Dimensions of Prepositions and their Use in Computational Linguistic Formalisms and Applications*.

Kemmerer, D. (1999). "Near" and "far" in language and perception. *Cognition*, 73(1):35 – 63.

Kennedy, C. (2007). Vagueness and grammar: The semantics of relative and absolute gradable adjectives. *Linguistics and Philosophy*, 30(1):1–45.

Kiczales, G., Des Rivieres, J., and Bobrow, D. (1991). *The art of the metaobject protocol*. The MIT Press.

Kirby, S. and Hurford, J. R. (2002). The emergence of linguistic structure: An overview of the iterated learning model. In *Simulating the evolution of language*, pages 121–147. Springer.

Klein, E. (1980). A semantics for positive and comparative adjectives. *Linguistics and philosophy*, 4(1):1–45.

Krifka, M. (2007). Approximate interpretation of number words: A case for strategic communication. *Cognitive foundations of interpretation*, pages 111–126.

Krueger, L. E. (1972). Perceived numerosity. *Perception & Psychophysics*, 11(1A).

Kurtzman, H. and MacDonald, M. (1993). Resolution of quantifier scope ambiguities. *Cognition*, 48(3):243–279.

Lakoff, G. (1973). Hedges: A study in meaning criteria and the logic of fuzzy concepts. *Journal of philosophical logic*, 2(4):458–508.

Lakoff, G. (1987). *Women, Fire, and Dangerous Things: What Categories Reveal about the Mind*. University of Chicago Press.

Langacker, R. W. (1987). *Foundations of Cognitive Grammar. Volume 1*. Stanford University Press, Stanford.

Lappin, S. (2000). An intensional parametric semantics for vague quantifiers. *Linguistics and philosophy*, 23(6):599–620.

Levinson, S. C. (1996). Language and space. *Annual review of Anthropology*, 25(1):353–382.

Lewis, D. (1970). General semantics. *Synthese*, 22(1):18–67.

Lloyd, S. P. (1982). Least squares quantization in PCM. *IEEE Transactions on Information Theory*, 28(2):129–137.

Mainwaring, S., Tversky, B., Ohgishi, M., and Schiano, D. (2003). Descriptions of simple spatial scenes in English and Japanese. *Spatial Cognition and Computation*, 3(1):3–42.

Mandler, G. and Shebo, B. (1982). Subitizing: An analysis of its component processes. *Journal of Experimental Psychology: General*, 111(1):1.

Manning, C., Raghavan, P., and Schütze, H. (2008). *Introduction to information retrieval*. Cambridge University Press.

Marmasse, N., Bletsas, A., and Marti, S. (2000). Numerical mechanisms and childrens concept of numbers. *The Media Laboratory Massachusetts Institute of Technology*, 20.

Mehler, J. and Bever, T. G. (1967). Cognitive capacity of very young children. *Science*.

Minturn, A. and Reese, T. (1951). The effect of differential reinforcement on the discrimination of visual number. *The Journal of Psychology*, 31(2):201–231.

Mitchell, T. M. (1997). *Machine learning*. McGraw Hill.

Mix, K., Huttenlocher, J., and Levine, S. (2002). Multiple cues for quantification in infancy: Is number one of them? *Psychological Bulletin*, 128(2):278–294.

Montague, R. (1974). The proper treatment of quantification in ordinary English. *Formal Semantics: The Essential Readings*.

Moxey, L. and Sanford, A. (1993a). *Communicating quantities: A psychological perspective.* Lawrence Erlbaum Associates, Inc.

Moxey, L. M. and Sanford, A. J. (1993b). Prior expectation and the interpretation of natural language quantifiers. *European Journal of Cognitive Psychology*, 5(1):73–91.

Newstead, S. and Coventry, K. (2000). The role of context and functionality in the interpretation of quantifiers. *European Journal of Cognitive Psychology*, 12(2):243–259.

Pauw, S. (2013a). Semiotic dynamics. In Steels, L., editor, *Grammaticalization of Gradable Quantifiers, from Adjective to Quantifier*. Forthcoming.

Pauw, S. (2013b). Size matters: Modeling the adjectival origin of gradable quantifiers.

Pauw, S. (2013c). Vagueness reader. In Veltman, F., editor, *The Adjectival Origins of Vague Quantifiers: A Cognitive Explanation*. Forthcoming.

Pauw, S. (submitted). Size matters: A cognitive semantics for quantity.

Pauw, S. and Hilfery, J. (2012). The emergence of quantifiers. In Steels, L., editor, *Experiments in Cultural Language Evolution*. John Benjamins.

Pauw, S. and Spranger, M. (2010). Embodied determiners. In Slavkovik, M., editor, *Proceedings of Language Evolution and Computation Workshop at ESSLLI*, pages 184–192.

Pauw, S. and Spranger, M. (2012). Embodied quantifiers. In Lassiter, D. and Slavkovik, M., editors, *New Directions in Logic, Language and Computation*, pages 52–66. Springer.

Piaget, J. (1952). *The child's conception of number*. Taylor & Francis.

Piaget, J. (1968). Quantification, conservation, and nativism. *Science*.

Pufall, P., Shaw, R., and Syrdal-Lasky, A. (1973). Development of number conservation: an examination of some predictions from piaget's stage analysis and equilibration model. *Child Development*, pages 21–27.

Pufall, P. B. and Shaw, R. E. (1972). Precocious thoughts on number: The long and the short of it. *Developmental Psychology; Developmental Psychology*, 7(1):62.

Roorda, A. and Williams, D. R. (1999). The arrangement of the three cone classes in the living human eye. *Nature*, 397:520–522.

Rosch, E. and Lloyd, B., editors (1978). *Cognition and categorization*. Lawrence Erlbaum Associates, Hillsdale, NJ.

Rosch, E., Mervis, C., Gray, W., Johnson, D., and Boyes-Braem, P. (2004). Basic objects in natural categories. *Cognitive Psychology: Key Readings*, page 448.

Sapir, E. (1944). Grading, a study in semantics. *Philosophy of Science*, 11(2):93–116.

Solt, S. (2009). *The semantics of adjectives of quantity*. PhD thesis, The City University of New York.

Sommerer, L. (2012). Investigating the emergence of the definite article in old english: About categorization, gradualness and constructions. *Folia Linguistica Historica*, 33(1):175–213.

Spranger, M. (2008). World models for grounded language games. German diplom (master) thesis, Humboldt-Universitaet zu Berlin.

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

Spranger, M. (2012). The co-evolution of basic spatial terms and categories. In Steels, L., editor, *Experiments in Cultural Language Evolution*. John Benjamins, Amsterdam.

Spranger, M. and Loetzsch, M. (2009). The semantics of sit, stand, and lie embodied in robots. In Taatgen, N. A. and van Rijn, H., editors, *Proceedings of the 31th Annual Conference of the Cognitive Science Society (Cogsci09)*, pages 2546–2552, Austin, TX. Cognitive Science Society.

Spranger, M. and Loetzsch, M. (2011). Syntactic Indeterminacy and Semantic Ambiguity: A Case Study for German Spatial Phrases. In Steels, L., editor, *Design Patterns in Fluid Construction Grammar*, volume 11 of *Constructional Approaches to Language*, pages 265–298. John Benjamins.

Spranger, M., Loetzsch, M., and Pauw, S. (2010a). Open-ended Grounded Semantics. In Coelho, H., Studer, R., and Woolridge, M., editors, *Proceedings of the 19th European Conference on Artificial Intelligence (ECAI 2010)*, pages 929–934, Amsterdam. IOS Press.

Spranger, M., Loetzsch, M., and Steels, L. (2012a). A Perceptual System for Language Game Experiments. In Steels, L. and Hild, M., editors, *Language Grounding in Robots*. Springer, New York.

Spranger, M. and Pauw, S. (2012). Dealing with perceptual deviation: Vague semantics for spatial language and quantification. In Steels, L. and Hild, M., editors, *Language Grounding in Robots*, pages 173–192. Springer, New York.

Spranger, M., Pauw, S., and Loetzsch, M. (2010b). Open-ended semantics co-evolving with spatial language. *The Evolution of Language: Proceedings of the 8th International Conference (EVOLANG8)*, page 297.

Spranger, M., Pauw, S., Loetzsch, M., and Steels, L. (2012b). Open-ended Procedural Semantics. In Steels, L. and Hild, M., editors, *Language Grounding in Robots*. Springer, New York.

Spranger, M. and Steels, L. (2012). Emergent Functional Grammar for Space. In Steels, L., editor, *Experiments in Cultural Language Evolution*. John Benjamins, Amsterdam.

Starkey, P. and Cooper, R. (1980). Perception of numbers by human infants. *Science*, 210(4473):1033.

Starkey, P., Spelke, E., and Gelman, R. (1990). Numerical abstraction by human infants. *Cognition*.

Starkey, P., Spelke, E. S., Gelman, R., et al. (1983). Detection of intermodal numerical correspondences by human infants. *Science*, 222(4620):179–181.

Stechow, A., CRESSWELL, M., and HELLAN, L. (1984). Comparing semantic theories of comparison. *Journal of semantics*, 3(1-2):1–92.

Steedman, M. and Baldridge, J. (2009). Combinatory categorial grammar. *Non-transformational Syntax: A Guide to Current Models. Blackwell, Oxford*.

Steels, L. (1982). Constraints as consultants. In *Proceedings of the European Conference on Artificial Intelligence (ECAI)*, pages 75–78, Orsay, France.

Steels, L. (1999). The talking heads experiment.

Steels, L. (2000a). The emergence of grammar in communicating autonomous robotic agents. In Horn, W., editor, *Proceedings of the 14th European Conference on Artificial Intelligence (ECAI)*, pages 764–769, Berlin, Germany. IOS Press.

Steels, L. (2000b). The emergence of grammar in communicating autonomous robotic agents. *ECAI*, pages 764–769.

Steels, L. (2001). Language games for autonomous robots. *IEEE Intelligent systems*, pages 16–22.

Steels, L. (2003). Language re-entrance and the 'inner voice'. *Journal of Consciousness Studies*, 10(4-5):173–185.

Steels, L. (2005). The emergence and evolution of linguistic structure: from lexical to grammatical communication systems. *Connection science*, 17(3-4):213–230.

Steels, L. (2008). The symbol grounding problem has been solved. so whats next. *Symbols and embodiment: Debates on meaning and cognition*, pages 223–244.

Steels, L. (2012a). Exploring cultural language evolution with language games. In Steels, L., editor, *Experiments in Cultural Language Evolution*. John Benjamins, Amsterdam.

Steels, L. (2012b). Grounding Language through Evolutionary Language Games. In Steels, L. and Hild, M., editors, *Language Grounding in Robots*. Springer, New York.

Steels, L. (2012c). Self-organization and selection in cultural language evolution. In Steels, L., editor, *Experiments in Cultural Language Evolution*. John Benjamins, Amsterdam.

Steels, L. and Bleys, J. (2005). Planning what to say: Second order semantics for fluid construction grammars. In *Proceedings of CAEPIA*, volume 5.

Steels, L. and De Beule, J. (2006). Unify and merge in Fluid Construction Grammar. In Vogt, P., Sugita, Y., Tuci, E., and Nehaniv, C., editors, *Symbol Grounding and Beyond: Proceedings of the Third International Workshop on the Emergence and Evolution of Linguistic Communication*, LNAI 4211, pages 197–223. Springer.

Steels, L., De Beule, J., and Wellens, P. (2012a). Fluid Construction Grammar on Real Robots. In Steels, L. and Hild, M., editors, *Language Grounding in Robots*. Springer, New York.

Steels, L. and Kaplan, F. (2002). AIBO's first words: The social learning of language and meaning. *Evolution of Communication*, 4(1):3–32.

Steels, L. and Loetzsch, M. (2009). Perspective alignment in spatial language. In Coventry, K. R., Tenbrink, T., and Bateman, J. A., editors, *Spatial Language and Dialogue*, pages 70–89. Oxford University Press.

Steels, L. and Spranger, M. (2008). The robot in the mirror. *Connection Science*, 20(4):337–358.

Steels, L. and Spranger, M. (2012). Emergent mirror systems for body language. In Steels, L., editor, *Experiments in Cultural Language Evolution*, pages 87—109. John Benjamins.

Steels, L., Spranger, M., van Trijp, R., Höfer, S., and Hild, M. (2012b). Emergent action language on real robots. In Steels, L. and Hild, M., editors, *Language Grounding in Robots*, chapter 13. Springer, New York.

Steels, L. and Wellens, P. (2006). How grammar emerges to dampen combinatorial search in parsing. *Symbol grounding and beyond*, pages 76–88.

Strauss, M. S. and Curtis, L. E. (1981). Infant perception of numerosity. *Child Development*, pages 1146–1152.

Sussman, G. and Steele, G. (1980). Constraints - a language for expressing almost-hierarchical descriptions. *Artif. Intell.*, 14(1):1–39.

Taves, E. (1941). *Two mechanisms for the perception of visual numerousness*. Number 259-266. Columbia University.

Taylor, H. A. and Tversky, B. (1996). Perspective in Spatial Descriptions. *Journal of Memory and Language*, 35(3):371–391.

Tenbrink, T. (2005). Identifying objects on the basis of spatial contrast: an empirical study. *Spatial Cognition IV. Reasoning, Action, and Interaction*, pages 124–146.

Tenbrink, T. (2007). *Space, time, and the use of language: An investigation of relationships*, volume 36 of *Cognitive Linguistics Research*. Walter de Gruyter, Berlin, DE.

Tenbrink, T. and Moratz, R. (2003). Group-based spatial reference in linguistic human-robot interaction. In *Proceedings of EuroCogSci'03, The European Cognitive Science Conference 2003*, pages 325–330. Lawrence Erlbaum.

Tomasello, M. (2003). *Constructing a Language: A Usage-Based Theory of Language Acquisition*. Harvard University Press.

Trick, L. and Pylyshyn, Z. (1991). *A theory of enumeration that grows out of a general theory of vision: Subitizing, counting and FINSTs*. University of Western Ontario, Centre for Cognitive Science.

Trick, L. and Pylyshyn, Z. (1993). What enumeration studies can show us about spatial attention: Evidence for limited capacity preattentive processing. *Journal of Experimental Psychology*, 19(2):331–351.

Van Den Broeck, W. (2008). Constraint-based compositional semantics. In D.M. Smith, A., Smith, K., and Ferrer i Cancho, R., editors, *Proceedings of the 7th International Conference on the Evolution of Language (EVOLANG 7)*, pages 338–345, Singapore. World Scientific Publishing.

van Eijck, J. (1990). *Discourse representation theory*. Centrum voor Wiskunde en Informatica.

van Rooij, R. (2006). Evolutionary games and semantic universals. In *The evolution of language: proceedings of the 6th international conference (EVOLANG6), Rome, Italy, 12-15 April 2006*, page 356. World Scientific Pub Co Inc.

van Rooij, R. (2011). Vagueness and linguistics. *Vagueness: A guide*, pages 123–170.

van Trijp, R. (2008). *Analogy and Multi-Level Selection in the Formation of a Case Grammar. A Case Study in Fluid Construction Grammar*. PhD thesis, Universiteit Antwerpen.

van Trijp, R. (2012). The emergence of case marking systems for marking event structure. In Steels, L., editor, *Experiments in Cultural Language Evolution*. John Benjamins, Amsterdam.

van Trijp, R. (2012). Self-assessing agents for explaining language change: A case study in german. In *Proceedings of ECAI 2012*.

von Fintel, K. and Matthewson, L. (2007). Universals in Semantics? *The Linguistic Review*, 25(1–2):139–201.

Wilson, D. and Sperber, D. (1993). Linguistic form and relevance. *Lingua*.

Winograd, T. (1971). *Procedures as a representation for data in a computer program for understanding natural language*. PhD thesis, MIT.

Winograd, T. (1975). Frame representations and the declarative/procedural controversy. *Representation and understanding: Studies in cognitive science*, pages 185–210.

Winograd, T. (1980). What does it mean to understand language? *Cognitive science*, 4(3):209–241.

Wohlwill, J. and Lowe, R. (1962). Experimental analysis of the development of the conservation of number. *Child Development*, 33(1):153–167.

Woods, W. A. (1981). Procedural semantics as a theory of meaning. In Joshi, A. K., Webber, B. L., and Sag, I. A., editors, *Elements of Discourse Understanding*, pages 300–334. Cambridge University Press.

Wright, C. (1975). On the coherence of vague predicates. *Synthese*, 30(3):325–365.

Zadeh, L. (1965). Fuzzy sets. *Information and control*, 8(3):338–353.

Zadeh, L. A. (1983). A computational approach to fuzzy quantifiers in natural languages. *Computers & Mathematics with Applications*, 9(1):149–184.

# Index

# Samenvatting

Woorden zoals het Engelse *many* [*veel*] en *few* [*weinig*] zijn dualistisch van aard: hoewel traditionele analyses ze als kwantoor beschouwen ("many of the houses" ["veel van de huizen"]), gedragen ze zich ook als bijvoeglijke naamwoorden ("few /fewer houses" ["weinig/minder huizen"]). Feitelijk vallen dit soort termen syntactisch en semantisch samen met zowel kwantoren als bijvoeglijke naamwoorden. Waarom zijn ze niet ingeperkt tot één grammaticale klasse? Wat is de cognitieve basis voor hun dualistisch gedrag. En, hoe zou dit conceptuele en taalkundige dualisme hebben kunnen evolueren?

Historisch bewijs doet vermoeden dat de dualistische syntaxis van deze termen (van nu af, *gradable quantifiers* [graduele kwantoren]) het gevolg zou kunnen zijn van een grammaticalisatieproces, waarbij ze onstaan als bijvoeglijke naamwoorden en later in kwantoren veranderen. Een bijvoorbeeld is de Engelse kwantoor *few*, dat afgeleid is van het Oudengelse bijvoeglijke naamwoord *feawe*. Dit proefschrift onderzoekt de hypothese dat dit grammaticalisatiepad het gevolg zou kunnen zijn van de cognitieve relatie tussen omvang en aantal. Het schatten van het aantal objecten in een verzameling (een cognitief proces dat ten grondslag ligt aan termen zoals *few* en *many*) hangt af van een combinatie van perceptuele kenmerken zoals de omvang en de dichtheid van de verzameling. Sommige van deze kenmerken (zoals omvang) liggen ook ten grondslag aan predicaten als *big* [*groot*] en *small* [*klein*] en sommige van deze kenmerken (zoals dichtheid), zijn exclusief van toepassing op verzamelingen. Deze *cognitieve overlap* tussen de concepten omvang en aantal zou de dualiteit van *gradable quantifiers* kunnen verklaren: het verband met omvang motiveert het bijvoeglijk gebruik, terwijl de exclusieve toepassing op verzamelingen het kwantificationele gebruik motiveert.

Dit proefschrift beschrijft een reeks experimenten die de bovengenoemde hypothese onderzoekt in het kader van evolutionaire *language games* [*taalspellen*], waarbij robots zelf beschrijvingen ontwikkelen voor objecten (of, in dit geval, verzamelingen van objecten) in hun waargenomen omgeving. Dit model maakt het mogelijk de specifieke voorwaardes te ontdekken waaronder de hypothese

stand houdt. Door robots toe te rusten met een mechanisme dat hen in staat stelt aantallen in te schatten aan de hand van de kenmerken omvang en aantal, laat ik zien dat zij inderdaad termen ontwikkelen met de geobserveerde dualistische eigenschappen.

Een eerste reeks experimenten laat zien dat *gradable quantifiers* inderdaad ontstaan als bijvoeglijke naamwoorden door hun cognitieve overlap met omvangpredicaten zoals *big* en *small*. Een tweede reeks experimenten beschouwt de cognitieve overlap tussen *gradable quantifiers* en kwantoren. Deze experimenten laten zien dat deze cognitieve overlap ervoor zorgt dat de *gradable quantifiers* als kwantoren grammaticaliseren. Alles tezamen laat dit proefschrift zien dat de syntactische dualiteit van *gradable quantifiers* een onderliggende cognitieve dualiteit weerspiegelt. In algemenere zin maakt dit proefschrift duidelijk hoe het in acht nemen van cognitieve beperkingen inzicht kan geven in conceptuele en taalkundige dualiteit.

# Abstract

Words like *many* and *few* have a dual nature: though traditionally analyzed as quantifiers ("many of the houses"), they also behave like gradable adjectives ("few/fewer houses"). In fact, such terms pattern syntactically and semantically with both quantifiers and adjectives. Why arent they confined to one grammatical class? What is the cognitive basis for their dual behavior? And how might such conceptual and linguistic duality have evolved?

Historical evidence suggests that the dual syntax of these terms (henceforth, *gradable quantifiers*) might be the result of a grammaticalization process where they originate as adjectives and later become quantifiers, as illustrated by the quantifier *few*, based on the Old English adjective *feawe*. This dissertation explores the hypothesis that this grammaticalization path might be the result of the cognitive relationship between size and number. Judgments of size (underlying modifiers such as *big* and *small*) depend on perceptual features of objects (or sets of objects) in the environment. Judgments of approximate number (underlying terms like *few* and *many*) exploit a combination of spatial features that apply exclusively to sets of objects, such as their size and density. This *cognitive overlap* between the concepts of size and number may account for the duality observed in gradable quantifiers: the dependence on size motivates their adjectival uses, while their exclusive application to sets of objects motivates their quantificational uses.

This dissertation describes a series of experiments that captures the insight above within an evolutionary language games framework, in which robotic agents self-organize the means for describing objects (or in this case, groups of objects) in their perceived environment. The model allows the exploration of the specific conditions under which the hypothesis might hold. In particular, agents equipped with an approximate number sense that incorporates size can be shown to develop linguistic terms with the dual functions observed in gradable quantifiers.

A first set of experiments show that gradable quantifiers are indeed likely to emerge as adjectives due to their cognitive overlap with size predicates such as

*big* and *small*. But, gradable quantifiers are not only cognitively related to size predicates but also to quantifiers such as *all* or *three* (they eclusively apply to sets). A second set of experiments show that this cognitive overlap will invite the (initially adjectival) gradable quantifiers to grammaticalize into quantifiers. Overall, this dissertation shows that the syntactic duality of gradable quantifiers might be reflecting an underlying cognitive duality. More generally, it suggests how the inclusion of cognitive constraints may illuminate the origins of both conceptual and linguistic duality.

*Titles in the ILLC Dissertation Series:*

ILLC DS-2006-01: **Troy Lee**
*Kolmogorov complexity and formula size lower bounds*

ILLC DS-2006-02: **Nick Bezhanishvili**
*Lattices of intermediate and cylindric modal logics*

ILLC DS-2006-03: **Clemens Kupke**
*Finitary coalgebraic logics*

ILLC DS-2006-04: **Robert Špalek**
*Quantum Algorithms, Lower Bounds, and Time-Space Tradeoffs*

ILLC DS-2006-05: **Aline Honingh**
*The Origin and Well-Formedness of Tonal Pitch Structures*

ILLC DS-2006-06: **Merlijn Sevenster**
*Branches of imperfect information: logic, games, and computation*

ILLC DS-2006-07: **Marie Nilsenova**
*Rises and Falls. Studies in the Semantics and Pragmatics of Intonation*

ILLC DS-2006-08: **Darko Sarenac**
*Products of Topological Modal Logics*

ILLC DS-2007-01: **Rudi Cilibrasi**
*Statistical Inference Through Data Compression*

ILLC DS-2007-02: **Neta Spiro**
*What contributes to the perception of musical phrases in western classical music?*

ILLC DS-2007-03: **Darrin Hindsill**
*It's a Process and an Event: Perspectives in Event Semantics*

ILLC DS-2007-04: **Katrin Schulz**
*Minimal Models in Semantics and Pragmatics: Free Choice, Exhaustivity, and Conditionals*

ILLC DS-2007-05: **Yoav Seginer**
*Learning Syntactic Structure*

ILLC DS-2008-01: **Stephanie Wehner**
*Cryptography in a Quantum World*

ILLC DS-2008-02: **Fenrong Liu**
*Changing for the Better: Preference Dynamics and Agent Diversity*

ILLC DS-2008-03: **Olivier Roy**
*Thinking before Acting: Intentions, Logic, Rational Choice*

ILLC DS-2008-04: **Patrick Girard**
*Modal Logic for Belief and Preference Change*

ILLC DS-2008-05: **Erik Rietveld**
*Unreflective Action: A Philosophical Contribution to Integrative Neuroscience*

ILLC DS-2008-06: **Falk Unger**
*Noise in Quantum and Classical Computation and Non-locality*

ILLC DS-2008-07: **Steven de Rooij**
*Minimum Description Length Model Selection: Problems and Extensions*

ILLC DS-2008-08: **Fabrice Nauze**
*Modality in Typological Perspective*

ILLC DS-2008-09: **Floris Roelofsen**
*Anaphora Resolved*

ILLC DS-2008-10: **Marian Counihan**
*Looking for logic in all the wrong places: an investigation of language, literacy and logic in reasoning*

ILLC DS-2009-01: **Jakub Szymanik**
*Quantifiers in TIME and SPACE. Computational Complexity of Generalized Quantifiers in Natural Language*

ILLC DS-2009-02: **Hartmut Fitz**
*Neural Syntax*

ILLC DS-2009-03: **Brian Thomas Semmes**
*A Game for the Borel Functions*

ILLC DS-2009-04: **Sara L. Uckelman**
*Modalities in Medieval Logic*

ILLC DS-2009-05: **Andreas Witzel**
*Knowledge and Games: Theory and Implementation*

ILLC DS-2009-06: **Chantal Bax**
*Subjectivity after Wittgenstein. Wittgenstein's embodied and embedded subject and the debate about the death of man.*

ILLC DS-2009-07: **Kata Balogh**
*Theme with Variations. A Context-based Analysis of Focus*

ILLC DS-2009-08: **Tomohiro Hoshi**
*Epistemic Dynamics and Protocol Information*

ILLC DS-2009-09: **Olivia Ladinig**
*Temporal expectations and their violations*

ILLC DS-2009-10: **Tikitu de Jager**
*"Now that you mention it, I wonder…": Awareness, Attention, Assumption*

ILLC DS-2009-11: **Michael Franke**
*Signal to Act: Game Theory in Pragmatics*

ILLC DS-2009-12: **Joel Uckelman**
*More Than the Sum of Its Parts: Compact Preference Representation Over Combinatorial Domains*

ILLC DS-2009-13: **Stefan Bold**
*Cardinals as Ultrapowers. A Canonical Measure Analysis under the Axiom of Determinacy.*

ILLC DS-2010-01: **Reut Tsarfaty**
*Relational-Realizational Parsing*

ILLC DS-2010-02: **Jonathan Zvesper**
*Playing with Information*

ILLC DS-2010-03: **Cédric Dégremont**
*The Temporal Mind. Observations on the logic of belief change in interactive systems*

ILLC DS-2010-04: **Daisuke Ikegami**
*Games in Set Theory and Logic*

ILLC DS-2010-05: **Jarmo Kontinen**
*Coherence and Complexity in Fragments of Dependence Logic*

ILLC DS-2010-06: **Yanjing Wang**
*Epistemic Modelling and Protocol Dynamics*

ILLC DS-2010-07: **Marc Staudacher**
*Use theories of meaning between conventions and social norms*

ILLC DS-2010-08: **Amélie Gheerbrant**
*Fixed-Point Logics on Trees*

ILLC DS-2010-09: **Gaëlle Fontaine**
*Modal Fixpoint Logic: Some Model Theoretic Questions*

ILLC DS-2010-10: **Jacob Vosmaer**
*Logic, Algebra and Topology. Investigations into canonical extensions, duality theory and point-free topology.*

ILLC DS-2010-11: **Nina Gierasimczuk**
*Knowing One's Limits. Logical Analysis of Inductive Inference*

ILLC DS-2010-12: **Martin Mose Bentzen**
*Stit, Iit, and Deontic Logic for Action Types*

ILLC DS-2011-01: **Wouter M. Koolen**
*Combining Strategies Efficiently: High-Quality Decisions from Conflicting Advice*

ILLC DS-2011-02: **Fernando Raymundo Velazquez-Quesada**
*Small steps in dynamics of information*

ILLC DS-2011-03: **Marijn Koolen**
*The Meaning of Structure: the Value of Link Evidence for Information Retrieval*

ILLC DS-2011-04: **Junte Zhang**
*System Evaluation of Archival Description and Access*

ILLC DS-2011-05: **Lauri Keskinen**
*Characterizing All Models in Infinite Cardinalities*

ILLC DS-2011-06: **Rianne Kaptein**
*Effective Focused Retrieval by Exploiting Query Context and Document Structure*

ILLC DS-2011-07: **Jop Briët**
*Grothendieck Inequalities, Nonlocal Games and Optimization*

ILLC DS-2011-08: **Stefan Minica**
*Dynamic Logic of Questions*

ILLC DS-2011-09: **Raul Andres Leal**
*Modalities Through the Looking Glass: A study on coalgebraic modal logic and their applications*

ILLC DS-2011-10: **Lena Kurzen**
*Complexity in Interaction*

ILLC DS-2011-11: **Gideon Borensztajn**
*The neural basis of structure in language*

ILLC DS-2012-01: **Federico Sangati**
*Decomposing and Regenerating Syntactic Trees*

ILLC DS-2012-02: **Markos Mylonakis**
*Learning the Latent Structure of Translation*

ILLC DS-2012-03: **Edgar José Andrade Lotero**
*Models of Language: Towards a practice-based account of information in natural language*

ILLC DS-2012-04: **Yurii Khomskii**
*Regularity Properties and Definability in the Real Number Continuum: idealized forcing, polarized partitions, Hausdorff gaps and mad families in the projective hierarchy.*

ILLC DS-2012-05: **David García Soriano**
*Query-Efficient Computation in Property Testing and Learning Theory*

ILLC DS-2012-06: **Dimitris Gakis**
*Contextual Metaphilosophy - The Case of Wittgenstein*

ILLC DS-2012-07: **Pietro Galliani**
*The Dynamics of Imperfect Information*

ILLC DS-2012-08: **Umberto Grandi**
*Binary Aggregation with Integrity Constraints*

ILLC DS-2012-09: **Wesley Halcrow Holliday**
*Knowing What Follows: Epistemic Closure and Epistemic Logic*

ILLC DS-2012-10: **Jeremy Meyers**
*Locations, Bodies, and Sets: A model theoretic investigation into nominalistic mereologies*

ILLC DS-2012-11: **Floor Sietsma**
*Logics of Communication and Knowledge*

ILLC DS-2012-12: **Joris Dormans**
*Engineering emergence: applied theory for game design*