Technical Report

UCAM-CL-TR-663
ISSN 1476-2986

Number 663

UNIVERSITY OF CAMBRIDGE

Computer Laboratory

Syntax-driven analysis of context-free languages with respect to fuzzy relational semantics

Richard Bergmair

March 2006

15 JJ Thomson Avenue
Cambridge CB3 0FD
United Kingdom
phone +44 1223 763500

http://www.cl.cam.ac.uk/

Technical reports published by the University of Cambridge
Computer Laboratory are freely available via the Internet:

http://www.cl.cam.ac.uk/TechReports/

ISSN 1476-2986

Syntax-driven analysis of context-free languages with respect to fuzzy relational semantics

Richard Bergmair

Abstract

A grammatical framework is presented that augments context-free production rules with semantic production rules that rely on fuzzy relations as representations of fuzzy natural language concepts. It is shown how the well-known technique of syntax-driven semantic analysis can be used to infer from an expression in a language defined in such a semantically augmented grammar a weak ordering on the possible worlds it describes. Considering the application of natural language query processing, we show how to order elements in the domain of a relational database scheme according to the degree to which they fulfill the intuition behind a given natural language statement like `Carol lives in a small city near San Francisco`.

1 Introduction and motivation

It may be one of the most well established traditions in science to view an expression, no matter whether drawn from a formal or a natural language, as a strict binary decision boundary making a sharp distinction between that which is fundamentally right and that which is fundamentally wrong. However, for a large number of application areas, maybe even the majority of them, this principle does not reflect any conceivable reality. When Zadeh (1965) first stated that, in complete contradiction to the well-established dictum, "the classes of objects encountered in the real world do not have precisely defined criteria of membership" he probably expressed what was on the minds of most practitioners working in any of those areas at the time.

> "I'd like my new heating to heat up *slightly* if the heat isn't already *quite high* and the outside temperature is *fairly low*." — "I'd like my new camera to use the flashlight only if the lighting is *really low* and the camera detects that I can't hold the camera *steadily* for *long enough*." – "I'd like that database query to give me all students who receive a *high* bursary and spend *way too much* money on clothes."

How is an engineer going to translate those specifications into the world of binary decision boundaries? Is it okay if the heating-controller increases the target temperature by 2 °C (instead of *"slightly"*), if the temperature isn't already > 24 °C (instead of *"quite high"*) and the outside temperature is < 10 °C (instead of *"really low"*)? If those decision boundaries perfectly match the intuition behind the original specification, would it be

fair to say that the parameters 2.2 °C, > 23 °C, and < 11 °C violate this same intuition, just because some threshold had to be fixed when designing that heating controller?

> In a consumer-satisfaction survey, 65% of the respondents claimed to feel *extremely* refreshed when drinking Frizz-cola. Another 20% claimed to feel *very* refreshed. Only 10% said they would feel *not very* refreshed, and as little as 5% said they would feel *not* refreshed *at all*.

What is a statistician going to do with this data to conclude, whether or not Frizz-Cola is refreshing? Does a respondent who calls Frizz-cola *extremely* refreshing make up for twice as much evidence to conclude that it is, than a respondent who calls it *very* refreshing? If 2 is precisely the correct factor, would it not be absurd to assume that 2.05 is not, just because a linear scale had to be used, so traditional statistic tools could be applied?

- The car that the man has is nice.
- The wheel that the car that the man has has is nice.
- The rim that the wheel that the car that the man has has has is nice.

Which of these sentences should a linguist accept as grammatically well-formed? Would it be fair to say, that an "onion-sentence" like this may not use more than 2 levels of centre embedding? If an onion-sentence with 2 shells is absolutely grammatical, would it be okay to refuse a sentence with 3 shells as entirely ungrammatical, just because such a judgement had to be made to write a grammar?

In this paper, we will follow up on Lakoff's insight that "Natural Language concepts are fuzzy" and that "therefore natural logic must be a fuzzy logic" (see Parret 1974, p. 196). We will give up on strict logical distinctions between that which is fundamentally right and that which is fundamentally wrong, and, like Zadeh did, talk about degrees to which something is right or wrong. We will give up on strict syntactic distinctions between that which is absolutely grammatical and that which is absolutely ungrammatical, and, like Lakoff did, talk about degrees to which something is grammatical or ungrammatical. The primary objective of our model of natural language semantics is not to arrive at a partitioning of possible worlds described by the statement but rather at an ordering of them. We believe that this approach makes sense in the context of most modern applications of natural language technology, the most obvious example being information retrieval: A search engine that outputs an unordered set of search results would clearly not attract many users. This is why we believe that natural language semantics with respect to fuzzy models should be a highly rewarding object of study in natural language processing.

Furthermore we will follow up on the paradigm shift incepted by Zadeh when he introduced linguistic variables to fuzzy systems theory (Zadeh 1975*a*,*b*,*c*). We will not approach approximate reasoning as a mathematical toolset to manipulate sets in terms of arbitrarily chosen many-valued characteristic functions. Rather, we will try to reinforce fuzzy logic as the study of the inherently vague concepts of reasoning employed in human communication, and enforce the agreement of fuzzy models with our intuitions about the nature of the vague categories of reasoning they are meant to denote. We believe that this is what distinguishes fuzzy logic from other methods of soft computing such as neural networks, genetic algorithms, and computational learning theory, which tend to

turn their models into uninterpretable black-boxes. This is why we believe that fuzzy models as established by natural language semantics should be a rewarding object of study in fuzzy systems theory and approximate reasoning.

2 History of prior work

Zadeh's landmark paper (Zadeh 1965) is probably the starting point of fuzzy logic as we know it today. Although the importance of everyday quantitative expressions for the mathematical analysis of vague concepts was recognized earlier, for example by Sheppard (1954) in the domain of quantitative research methodology, Zadeh was the first to propose mathematical tools to cope with the general concept of fuzziness on a broad scale. At the time of their inception, Zadeh's ideas were well received in the systems engineering community, and developed into a rich toolset addressing the pressing needs of engineers to cope with imprecisely defined concepts. (See Gaines & Kohout (1977) for an exposition of early work in the field).

In the course of the first rush of euphoria, formal language theory, being part of the body of engineering wisdom that was generically made subject to "fuzzification" during that time, was taken into the fuzzy domain by Lee & Zadeh (1969). However the idea of a fuzzified formal language theory did not receive much attention, when fuzzy systems were only just beginning to be successfully applied to simple control-tasks and formal language theory was of interest only to fields like compiler construction that naturally had little use for vague concepts.

Linguists, on the other hand, remained widely unaware of the developments that took place in the engineering world, until Lakoff picked up the idea of Fuzzy Logic in the mid 1970s (see Parret 1974, p. 196). Unfortunately, although most linguists have had little trouble accepting the idea of vague concepts, its impact on semantic theory has remained only of secondary interest to linguists following the tradition of Chomsky (see Parret 1974, p. 50).

During the same period computer science saw the rise of artificial intelligence and its historically unparalleled interest in meaning representation. Considerable work on fuzzy meaning representation schemes was carried out by Goguen (1974), who also attempted to build a fuzzy SHRDLU, a robot capable of carrying out commands input in natural language in the domain of a fuzzy microworld (Goguen 1975). Zadeh proposed a fuzzy meaning representation scheme for natural languages as well (Zadeh 1978). However, these representation schemes were mainly concerned with meaning as such, rather than meaning in relation to natural languages. Later, Zadeh presented test-score semantics (Zadeh 1981, 1982) in an approach to bridge the gap between natural language representation and his fuzzy meaning representation. However his technique was never deployed in an actual grammar. A decade later, Novak presented what is probably the only work really concerned with the nuts and bolts of natural language from the point of view of fuzzy logic (Novak 1992, 1991).

Another important development of the mid 1970s was that, besides fuzzy logic, other methods of soft computing came along, such as neural networks and genetic algorithms. Fuzzy logic distinguished itself by putting renewed emphasis on the motivations that originally gave rise to its inception – the vaguely defined categories employed by humans in natural language reasoning. It was realized that humans use natural language expressions

to refer to those categories where fuzzy logic used sets defined in terms of numeric valued characteristic functions. This lead to the inception of the "linguistic variable" (Zadeh 1975*a*,*b*,*c*), a formal tool which makes explicit the correspondence between these two denotational variants of vague concepts. Engineers following the new paradigm were no longer free to pick fuzzy sets at will, but they had to bear in mind that fuzzy sets are meant to resemble meanings of natural language expressions. This aspect has only recently experienced renewed attention in an attempt to come to grips with what exactly it means for a linguistic variable to be interpretable in terms of natural language concepts (Bodenhofer & Bauer 2003, De Cock et al. 2000, De Cock & Kerre 2002).

After the early days of applying fuzzy logic to each and every conceivable problem that inspired the artificial intelligence community back in the 1970s, fuzzy logic experienced some decline in popularity in the decades that followed and matured to become the subject of major work on the foundations of mathematics, mainly carried out in eastern Europe, most notably the discovery of fuzzy logic as a generalization of classic logic that preserves its property of Hilbert completeness, and an operative technology that enabled many remarkable technical achievements, celebrated mostly by Japanese engineers. But, despite these successes it may be fair to say that, to this day, fuzzy logic has failed to live up to the high expectations artificial intelligence enthusiasts once had, when they set out to deploy the technology to make machines *understand* the categories of reasoning that humans use to successfully communicate to each other vague ideas and concepts.

Only recently, Zadeh took up renewed interest in this line of research, addressing the main shortcoming when he observes that "progress has been, and continues to be, slow in those areas where a methodology is needed in which the objects of computation are perceptions – perceptions of time, distance, form, direction, color, shape, truth, likelihood, intent, and other attributes of physical and mental objects" (Zadeh 2001). The key point Zadeh has to make about perceptions is that they are inherently fuzzy, and that humans use natural language representations where machines use numeric measurements. Thus, the paradigm shift that takes Zadeh into his "new direction of artificial intelligence" is one that takes us "from computing with numbers to computing with words" (Zadeh 1999). The representations of fuzzy concepts employed in his computational theory of perceptions are linguistic in nature. They are expressions of a language he refers to as precisiated natural language (Zadeh 2004*b*). Such a language would have to be natural, in the sense that it is a formal language weakly equivalent to a subset of a natural language, and precisiated, in the sense that every such expression can automatically be translated to a form suitable for approximate reasoning.

At this point we would like to highlight one rather questionable assumption underlying the more visionary end of Zadeh's ideas: that a reduction from the problem of "computing with words" to the strong AI problem is straightforward or even possible. For example Zadeh often cites applications such as parking a car, driving in city traffic, playing golf, or cooking a meal (Zadeh 2001) – those problems that actually do involve perceptions of time, distance, form, direction, color, or shape, and not just perceptions of language as such. His approach therefore presumes that representations of such perceptions in natural languages such as English or German pay justice to the actual objects of cognition, which assumes a flavour of Whorfianism possibly too strong for most contemporary linguists to savour.

Nevertheless, a technology as envisioned by Zadeh, that enables the computational

manipulation of linguistic expressions describing fuzzy concepts remains highly desirable. In fact, for the technology described in this paper in particular, we can think at least of two immediate applications: Natural language interfaces to flexible query processing systems (Zadeh 2003, 2004*a*, Dvorak & Novak 2000), and software tools supporting the implementation of fuzzy controllers in a linguistically intuitive way (Novak 1995, 1997, Bodenhofer & Bauer 2003). More remote applications of such technology may possibly include information extraction and retrieval and document classification.

3 The application: Natural language query answering

3.1 The use case

Imagine the following scenario: Police are investigating a case of bank robbery, in which one of the two perpetrators could be arrested, and confessed the identity of his accomplice. Unfortunately all he can hint out, is that the accomplice was introduced to him as "Carol", and that she had once mentioned that she lived in a small city near San Francisco. In order to find out where exactly Carol lives, police are now turning to the California registration office: It has available data about its citizens, its cities, the distances between cities and about where citizens reside in the form of a relational database, such as the one defined in Section 4.1.

The investigators get access to the standard querying system used by registration officers. It allows them to search for residency-records either by name or by city, and for cities by population or by their distances from other cities. Unfortunately none of these query masks is of any use for them, since there are too many people called "Carol" in California, too many small cities and too many cities near San Francisco.

They have to join forces with an IT specialist who formulates the necessary non-trivial SQL statements for them. Soon they find that not even SQL will provide the level of access they need. What is "a small city" in terms of population? What city is "near San Francisco" given its distance from San Francisco in kilometres? Ultimately they und up fixing some thresholds, and formulating a database query that returns all residency records, where Carol lives in a city whose population is less than 100000, and whose distance from San Francisco is less than 100 kilometres. Each of the vast number of records returned from that query in a random order is then checked by an investigator who assesses whether or not a given record could be relevant.

It is obvious that this is far from an optimal solution. What we will propose in this paper is a query interface that would let the investigators enter an expression from a query language such as the one defined in Section 5.1. Since the query language is a subset of English, they could enter something like `Carol lives in a small city near San Francisco` (Zadeh 2001). The system would now return a sequence of records automatically ranked according to the degree to which they fulfill the intuition most humans will have about this statement.

For example, there might be a database record about someone called Carol who lives in Half Moon Bay, a city which is both small, and near San Francisco, and a database record about someone called Carol living in New York. Here it is clear that the former

record deserves a higher ranking than the latter. However, if there is someone called Carol who lives in Inverness, a city that is further away from San Francisco than Half Moon Bay but much smaller this is not so clear. Here we have to find some way to weigh the two criteria. One reasonable way of obtaining these weights would be to rely on our intuition about the fuzzy concepts behind `near` and `small`. Clearly `Carol lives in a really tiny city fairly close to San Francisco` favors Inverness over Half Moon Bay, and `Carol lives in a rather small city extremely close to San Francisco` favors Half Moon Bay over Inverness. It might be reasonable to assume that humans also use these fuzzy concepts to encode the weights we are looking for. Intensifying modifiers such as `very`, or `extremely` and weakening modifiers such as `fairly`, or `more or less` are used quite frequently in natural language. However, in most traditional models of natural language semantics, these modifiers do not play the important role they should, simply because their semantic function does not fit into the picture of a natural language expression as a strict binary decision boundary.

Fortunately Fuzzy Logic equips us with the tools necessary to extract such information. These will be described in Section 6. With Fuzzy Logic and vague categories of reasoning in the back of our mind, we will then turn to the obvious metalinguistic problem at hand: How can a given expression from a context-free language be mechanically analyzed with regard to our semantic model. Our solution will employ the rather well-established technique of syntax-driven semantic analysis described in Section 7.

In Section 8 we will then put together the nuts and bolts of fuzzy semantics to arrive at a model allowing us to derive the meaning of certain natural language expressions with respect to relational data models. Appendix A will use all of this in a worked-out example to determine a sequence of records taken from our example database ordered in such a way that those records are listed first, that best satisfy the expression `Carol lives in a small city near San Francisco`. Appendix B gives a PROLOG program that does the same.

3.2 Ordering-based semantics

More generally, the technique we will propose operates on a set of data represented in some way $Dat = \{\mathtt{d}_1, \mathtt{d}_2, \mathtt{d}_3, \ldots, \mathtt{d}_n\}$. An expression is taken to be a constraint *Con* on these records. Such a constraint can be represented by an n-ary relation $Con \subseteq Dat^n$ on *Dat* for some n.

In standard query languages (such as SQL) the semantics of query expressions are taken to be partition based.

Definition 1. The *crisp* or *partition-based* semantics of a query expression, with respect to a set of records *Dat* is given by a unary relation $Con_c \subseteq Dat$ on *Dat*. (i.e. a subset Con_c of *Dat*)

Such a relation Con_c partitions *Dat* into a set *True* of records in *Dat* that fulfill the constraint, and a set *False* of records in *Dat* that do not fulfill the constraint. This can be seen by letting $True = Con_c$ and $False = Dat \setminus Con_c$ or vice-versa. Obviously everything is *True* or *False* ($True \cup False = Dat$), and nothing is both *True* and *False* ($True \cap False = \emptyset$).

In contrast to this crisp approach, we will take a fuzzy approach by taking the semantics of query expressions to be ordering-based:

Definition 2. The *fuzzy* or *ordering-based* semantics of a query expression, with respect to a set of records *Dat* is given by a binary relation $Con_f \subseteq Dat \times Dat$ on *Dat* which is reflexive, transitive and complete (i.e. a weak ordering on *Dat*).

Such a relation Con_f can establish a weak ordering on *Dat* such that $Con_f(\mathsf{d}_1, \mathsf{d}_2)$ iff d_1 satisfies the constraint "at least as well as" d_2. Since Con_f is a weak ordering we permit two elements d_1 and d_2 to have "the same rank", i.e. we permit that both $Con_f(\mathsf{d}_1, \mathsf{d}_2)$ and $Con_f(\mathsf{d}_2, \mathsf{d}_1)$ although $\mathsf{d}_1 \neq \mathsf{d}_2$.

Consequently the semantics of the unordered set of records

$$Dat = \{\mathsf{d}_1, \mathsf{d}_2, \mathsf{d}_3, \ldots, \mathsf{d}_n\}$$

with respect to a constraint Con_c is given by

$$(\{\mathsf{d} \mid \mathsf{d} \text{ fulfills } Con_c\}, \{\mathsf{d} \mid \mathsf{d} \text{ does not fulfill } Con_c\})$$

in the case of partition-based semantics and by

$$\langle \mathsf{d}_1, \mathsf{d}_2, \mathsf{d}_3, \ldots, \mathsf{d}_n \rangle \text{ such that } i < j \rightarrow \mathsf{d}_i \text{ fulfills } Con_f \text{ at least as well as } \mathsf{d}_j$$

in the case of ordering-based semantics.

Definition 3. Let Con_f denote the ordering-based semantics of an expression. The *d*-defuzzification of Con_f is given by $Con_c = \{d' \mid Con_f(d', d)\}$.

So, given the ordering-based semantics $\langle \mathsf{d}_1, \mathsf{d}_2, \mathsf{d}_3, \ldots, \mathsf{d}_n \rangle$, we can derive the partition based semantics $(\{\mathsf{d}_1, \mathsf{d}_2, \ldots, \mathsf{d}_i\}, \{\mathsf{d}_{i+1}, \mathsf{d}_{i+2}, \ldots, \mathsf{d}_n\})$ by simply fixing a d_i. Intuitively, departing from the fuzzy semantics for a constraint Con_f, we arrive at its crisp semantics Con_c by stating that everything that fulfills Con_f at least as well as d_i fulfills Con_c in the crisp sense, and everything else does not.

Lemma 1. *If Con_c represents the crisp semantics of an expression with respect to a set of records Dat, then there also exists Con_f, such that Con_f denotes the ordering-based semantics of the same expression with respect to Dat, and Con_c is the* d*-defuzzification of Con_f for some* d.

Proof. Let $True = Con_c$ and $False = Dat - Con_c$. Choose weak orderings Con_T on *True* and Con_F on *False* arbitrarily. Let $Con_f(\mathsf{d}_1, \mathsf{d}_2)$ iff one of the following applies:

- $d_1 \in True$, $d_2 \in True$, and $Con_T(d_1, d_2)$
- $d_1 \in False$, $d_2 \in False$, and $Con_F(d_1, d_2)$
- $d_1 \in True$, $d_2 \in False$

Since *True* is enumerable and weakly ordered, we can choose an element $d \in True$ such that $Con_T(d', d)$ for all $d' \in True$. It is easy to see that the *d*-defuzzification of Con_f is Con_c. □

So, given the partition based semantics $(\{\mathsf{d}_1, \mathsf{d}_2, \ldots, \mathsf{d}_i\}, \{\mathsf{d}_{i+1}, \mathsf{d}_{i+2}, \ldots, \mathsf{d}_n\})$, we can derive the ordering-based semantics $\langle \mathsf{d}_1, \mathsf{d}_2, \mathsf{d}_3, \ldots, \mathsf{d}_n \rangle$ by fixing an arbitrary ordering $\langle \mathsf{d}_1, \mathsf{d}_2, \ldots, \mathsf{d}_i \rangle$ on $\{\mathsf{d}_1, \mathsf{d}_2, \ldots, \mathsf{d}_i\}$ and an arbitrary ordering $\langle \mathsf{d}_{i+1}, \mathsf{d}_{i+2}, \ldots, \mathsf{d}_n \rangle$ on $\{\mathsf{d}_{i+1}, \mathsf{d}_{i+2}, \ldots, \mathsf{d}_n\}$. If we concatenate these sequences to give us the fuzzy semantics $\langle \mathsf{d}_1, \mathsf{d}_2, \mathsf{d}_3, \ldots, \mathsf{d}_n \rangle$, then its d_i-defuzzification will yield the original partition-based semantics again.

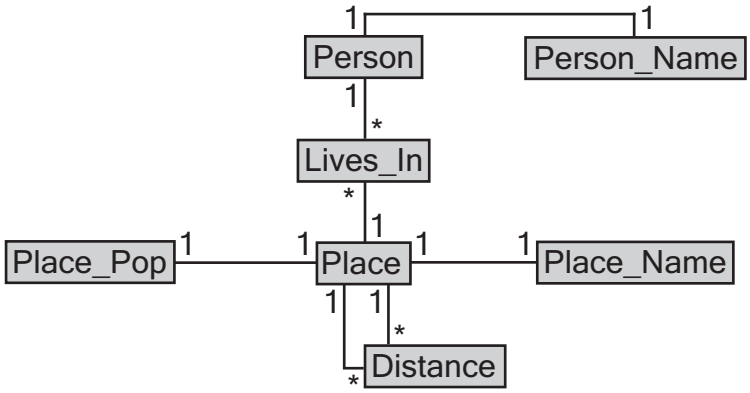


Figure 1: An entity-relationship diagram of our relational model

4 The semantic model: A relational database

4.1 An example data model

The domain we operate in is defined in terms of a crisp data model that allows us to reason about cities, about distances between cities, about people, and about who of the people lives in which of the cities.

Our scheme involves the following relations:

- Person($\underline{p}$): The primary key p refers to a person.
- Person_Name($\underline{p}, x$): The person referred to by primary key p has name x.
- Place($\underline{p}$): The primary key p refers to a place.
- Place_Name($\underline{p}, x$): The place referred to by primary key p has name x.
- Place_Pop($\underline{p}, x$): The place referred to by primary key p has population x.
- Place_Distance($\underline{p}, \underline{q}, x$): The places referred to by primary keys p and q are at a distance x from each other.
- Lives_In($\underline{x}, \underline{y}$): The tuple (x, y) is a primary key referring to a person x living in a place y.

In an industrial setup these relations will typically be database tables, or XML-files, but they might just as well be sets of PROLOG-facts, or data from a lexicon represented in some special format. Figure 1 shows an entity-relationship diagram and Figure 2 shows some example data for this relational scheme.

4.2 The relational model

More generally, we are interested in all semantic models that adhere to any relational scheme, i.e. any microworld domain that can be represented by any relational database.

Definition 4. A *relational scheme* $Sch = (Dom, Rel)$ consists of

Person
p_1
p_2
p_3
p_4

	Person_Name
p_1	Carol
p_2	Frank
p_3	Richard
p_4	John

Place
c_1
c_2
c_3
...
c_{57}

	Lives_In
p_1	c_{17}
p_2	c_{54}
p_3	c_{56}
p_4	c_{57}

	Place_Name
c_1	Altaneda
c_2	Antioch
c_3	Aptos
	...
c_6	Berkeley
	...
c_{17}	Half Moon Bay
	...
c_{28}	Oakland
	...
c_{39}	San Francisco
	...
c_{53}	New York
c_{54}	Los Angeles
c_{55}	Tokyo
c_{56}	Cambridge
c_{57}	Linz

	Place_Pop
c_1	43000
c_2	101124
c_3	9396
	...
c_6	102049
	...
c_{17}	12208
	...
c_{28}	398844
	...
c_{39}	751682
	...
c_{53}	8085742
c_{54}	3819951
c_{55}	12064100
c_{56}	131465
c_{57}	183504

		Place_Distance
c_1	c_2	537 km
c_1	c_3	460 km
		...
c_{39}	c_1	554 km
c_{39}	c_2	59 km
c_{39}	c_3	99 km
		...
c_{39}	c_6	16 km
		...
c_{39}	c_{17}	33 km
		...
c_{39}	c_{28}	13 km
		...
c_{39}	c_{39}	0 km
		...
c_{39}	c_{55}	8266 km
		...

Figure 2: Some example data for our database scheme

- A finite set Dom which establishes the data domain for all atomic values that appear throughout the database.
- A finite set Rel, of tuples $rel \in Rel$ of the form $rel = (R, n, p)$, where
  - $n \in \mathbb{N} \setminus \{0\}$ is the arity of the relation.
  - $R \subseteq Dom^n$ is an n-ary relation on Dom, i.e. R is a set of tuples of the form $(d_1, d_2, \ldots, d_n)$ where each $d_i \in Dom$.
  - $p \in \{1, 2, \ldots, n\}$ identifies a primary key, which is unique within a relation, i.e. if $r' \in R$ is of the form $r' = (r'_1, r'_2, \ldots, r'_n)$ and $r'' \in R$ is of the form $r'' = (r''_1, r''_2, \ldots, r''_n)$, then $r'_p = r''_p$ implies that $r' = r''$.

5 The syntactic model: A context-free language

5.1 An example grammar

Our query language is defined by a context-free grammar. As a point of departure note that we want the following expressions (`S`) to be in our language:

`S` → `Carol lives in a city near San Francisco`
`S` → `Carol lives in the large city near San Francisco`
`S` → `Carol lives in a very small city near San Francisco`
`S` → `Frank lives in San Francisco`

We observe that each of these sentences contains a verb (`V`).

`V` → `lives`

and some noun-phrases (`NP`).

`NP` → `Carol`.
`NP` → `San Francisco`.
`NP` → `a city near San Francisco`.

We can now redefine `S` by stating that `S` is of the form.

`S` → `NP V NP`

We observe that our new definition of `S` now contains a number of expressions that were not in our original definition of `S` such as `Frank lives in a very small city near San Francisco`. The fact that our new definition now generalizes to other expressions that perfectly match our intuition of what can be a query indicates that we are on the right track. Unfortunately it will also contain a number of expressions that were not in our original definition of `S` for a good reason, such as *`Frank lives Frank`. However, as

S → NP VP	PP → in NP
VP → V PP	AP → very AP
VP → V NP	Nom → Carol
NP → Nom	Nom → Frank
NP → Det N'	V → lives
N' → N	Det → a
N' → AP N	Adj → small
N' → N' PP	N → city
AP → Adj	Nom → San Fr.
PP → near NP	

Figure 3: Our example context-free grammar

we assume that we will use this grammar only for analysing sentences that are well-formed sentences of English, we can neglect this for the moment.

We could proceed with this kind of analysis, until we arrive at the the grammar given in Figure 5.1. At this point we would like to stress the fact that, from a linguistic point of view, this grammar is far too simplistic to actually describe a substantial fragment of English. However it serves demonstrative purposes quite well, as it is not entirely unreasonable from a linguistic point of view and readily accessible.

5.2 Context-free grammar

More generally, we are interested in all syntactic models that adhere to any context-free grammar.

Definition 5. A context-free grammar $G = (\mathtt{V}, \mathtt{T}, P, \mathtt{S})$ consists of

- A finite non-empty set $\mathtt{V}$ of *non-terminal* symbols.
- A finite non-empty set $\mathtt{T}$ of *terminal* symbols.
- A set P of production rules, each of which is of the form $\mathtt{A} \to \mathtt{a}_1\mathtt{a}_2 \ldots \mathtt{a}_\mathtt{n}$ where $\mathtt{A} \in \mathtt{V}$ is a grammar variable, and each $\mathtt{a}_i$ is a symbol, either terminal or non-terminal, i.e. $\mathtt{a}_i \in (\mathtt{V} \cup \mathtt{T})$.
- A *start-symbol* $\mathtt{S} \in \mathtt{V}$.

Definition 6. Fix a grammar $G = (\mathtt{V}, \mathtt{T}, P, \mathtt{S})$. We say that a non-terminal symbol $\mathtt{X} \in \mathtt{V}$ yields a string $\alpha = \mathtt{t}_1\mathtt{t}_2 \ldots \mathtt{t}_\mathtt{n}$ of terminals $\mathtt{t}_i \in T$ iff $\mathtt{X} \to \mathtt{x}_1\mathtt{x}_2 \ldots \mathtt{x}_\mathtt{m}$ is a production in P where for each $\mathtt{x}_i$

- $\mathtt{x}_i \in \mathtt{V}$ and $\mathtt{x}_i$ yields the string of terminals α_i, or
- $\mathtt{x}_i \in T$, in which case we take α_i to be $\mathtt{x}_i$

and α is the concatenation $\alpha_1 \cdot \alpha_2 \cdot \ldots \cdot \alpha_m$.

x	$height(x)$	$\mu_{TallMen}(x)$
joe	140 cm	0.00
hank	150 cm	0.00
steve	160 cm	0.25
john	170 cm	0.50
peter	180 cm	0.75
frank	210 cm	1.00
george	220 cm	1.00

Figure 4: Persons and heights in the example

6 The logic tool: Fuzzy sets and relations

6.1 An example for fuzzy reasoning

In order to work with fuzzy concepts, we will need to manipulate classes of objects with imprecisely defined criteria of membership, such as the class *TallMen* of tall men. The problem with the class of tall men, is that there is no sharp boundary deciding whether or not $john \in TallMen$, given that John's height is 170 cm. It would probably make sense to state that $frank \in TallMen$ if Frank's height is 210 cm, and that $hank \notin TallMen$ if Hank's height is 150 cm. The idea that lies at the basis of fuzzy logic is that everything in between will be a matter of degree. That is to say, although there may not be a an intuitive criterion by which to judge whether or not $john \in TallMen$ it will be easy to fix a ranking of tall men

$$\langle hank, john, frank \rangle.$$

Such a ranking can be constructed from a representation by means of numeric degrees of fulfillment from the unit interval. For example *frank*'s degree of membership in *TallMen* will be 1, when it does not make sense to state that *george* more closely resembles our intuition behind what constitutes a tall man than *frank*, just because George is even taller than Frank. Similarly, *hank*'s degree of membership in *TallMen* will be 0, when it does not make sense to state that *hank* more closely resembles our intuition behind what constitutes a tall man, than *joe*, just because Joe is even shorter than Hank. The degree of membership of *john* in *TallMen* will be 0.5, since we might want to assign a degree of membership 0.25 to *steve* to represent the fact that Steve's height is between that of Hank and that of John, and a degree of membership 0.75 to *peter* to represent the fact that Peter's height is between that of John and that of Frank.

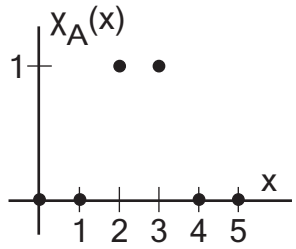
Given that, we can now rank the elements, by their degrees of fulfillment, and find that

$$\langle joe, hank, steve, john, peter, frank, george \rangle \text{ and}$$

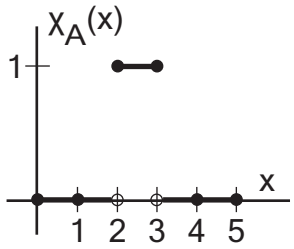
$$\langle hank, joe, steve, john, peter, george, frank \rangle$$

equally match the weak ordering established by those degrees of fulfillment and our intuition behind the class *TallMen*, while

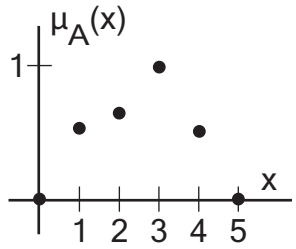
$$\langle joe, hank, peter, john, steve, frank, george \rangle$$



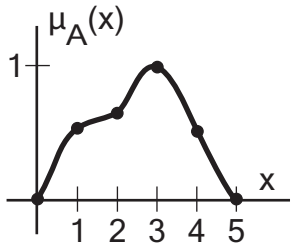
(a) crisp set on discrete universe



(b) crisp set on continuous universe



(c) fuzzy set on discrete universe



(d) fuzzy set on continuous universe

Figure 5: Characteristic functions of different kinds of sets

violates both.

6.2 Fuzzy sets

Fuzzy sets as proposed by Zadeh (1965) formally define these classes of objects with imprecisely defined criteria of membership. To establish fuzzy sets as a straightforward generalization of crisp sets, recall that $\mathcal{P}(X)$, denoting the powerset of X, is the set of all subsets $A \subseteq X$ of X. Recall that we can represent the set A in terms of a characteristic function as follows:

Definition 7. $A \in \mathcal{P}(X)$ is a crisp set on universe X, iff there exists a characteristic function $\chi_A : X \mapsto \{0, 1\}$ where

$$\chi_A(x) = \begin{cases} 1, & \text{if } x \in A \\ 0, & \text{if } x \notin A \end{cases}.$$

Now we can define fuzzy sets as a straightforward generalization of crisp sets, by letting their characteristic functions range over the whole unit interval.

Definition 8. (Zadeh 1965) $A \in \mathcal{F}(X)$ is a fuzzy set on universe X, iff there exists a characteristic function $\mu_B : X \mapsto [0, 1]$ that ranges over the whole unit interval where $\mu_A(x)$ is the degree to which x is a member in A.

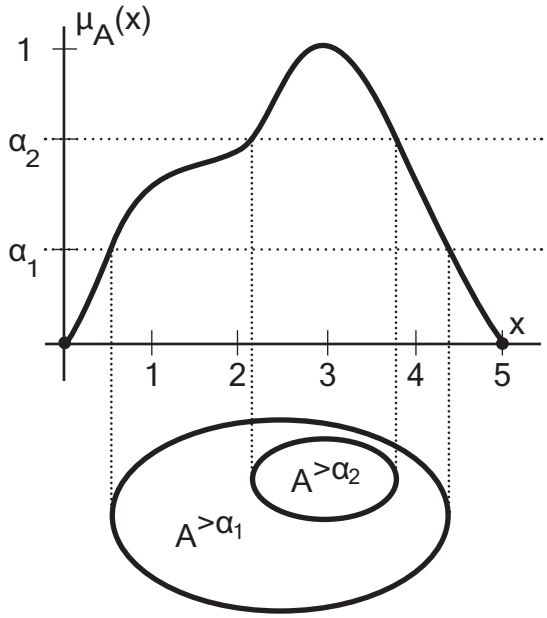


Figure 6: Alpha-cuts of fuzzy sets

To further justify the intuitive appeal of this approach, consider the notion of the α-cut.

Definition 9. The strict α-cut of a fuzzy set A is the crisp set

$$A^{>\alpha} = \{x | \mu_A(x) > \alpha\}.$$

Intuitively α measures something like the degree of strictness we may apply in the evaluation of fuzzy membership criteria when we approximate a fuzzy concept by a crisp set. That is to say, whenever we use a crisp set A' to denote a concept that is actually fuzzy, we implicitly make an arbitrary choice for α, and take A' to be $A^{>\alpha}$ for A being the actual fuzzy concept. Now let A_2 and A_1 be two crisp sets meant to approximate the same fuzzy concept A, the membership criteria of A_2 being evaluated at least as strictly as those of A_1, i.e. $A_2 = A^{>\alpha_2}$ and $A_1 = A^{>\alpha_1}$ with $\alpha_2 \geq \alpha_1$. By intuition we would now expect that $A_2 \subseteq A_1$, i.e. the membership of an element that can be verified on the basis of very strict criteria can also be verified on the basis of less strict criteria, but not necessarily vice-versa. More precisely:

Lemma 2. *For any fuzzy set A it is the case that for all α_1 and α_2*

$$\alpha_2 \geq \alpha_1 \Rightarrow A^{>\alpha_2} \subseteq A^{>\alpha_1}$$

Proof. We will show that for all x in the universe

$$x \in A^{>\alpha_2} \Rightarrow x \in A^{>\alpha_1}.$$

Note that, if $x \in A^{>\alpha_2}$, then, by definition 6, we have $\mu_A(x) > \alpha_2$. By hypothesis we have $\alpha_2 \geq \alpha_1$. From the well-ordering property of the unit-interval we have $\mu_A(x) > \alpha_1$. By definition 6 we have $x \in A^{>\alpha_1}$. □

6.3 Intersections of fuzzy sets

Given this notion of a fuzzy set we can extend the notion of set intersection from classical set theory into the fuzzy domain.

If $F \in \mathcal{F}(X)$ and $G \in \mathcal{F}(X)$ are two fuzzy sets on X, a fuzzy set $H \in \mathcal{F}(X)$ which is the intersection $H = F \cap_T G$ of F and G can now be defined in terms of the characteristic functions μ_H, μ_F and μ_G, more precisely:

Definition 10. (see Klement et al. 2000) A fuzzy set $H \in \mathcal{F}(X)$ is the intersection of two fuzzy sets $F \in \mathcal{F}(X)$ and $G \in \mathcal{F}(X)$ with respect to a triangular norm T, denoted $H = F \cap_T G$, iff $\mu_H(x) = T(\mu_F(x), \mu_G(x))$.

Zadeh's original choice of T was the minimum-function

$$T_{\min}(x, y) = \begin{cases} x, & \text{if } x \leq y \\ y, & \text{if } x > y \end{cases}.$$

In a first approach this choice is absolutely intuitive, considering that $T_{\min}(x, y) = x \wedge y$, where $\wedge$ is the crisp conjunction of two propositions x and y, that take values 0 and 1, where 0 is falsehood and 1 is truth.

More generally, triangular norms, are defined as follows:

Definition 11. (see Klement et al. 2000) A function $T : [0, 1] \times [0, 1] \mapsto [0, 1]$ is a triangular norm, iff it satisfies:

$$T(x, y) = T(y, x) \qquad \text{(commutativity)} \qquad (1)$$
$$T(x, T(y, z)) = T(T(x, y), z) \qquad \text{(associativity)} \qquad (2)$$
$$x \leq y \Rightarrow T(x, z) \leq T(y, z) \qquad \text{(non-decreasingness)} \qquad (3)$$
$$T(x, 1) = x \qquad \text{(neutral element)} \qquad (4)$$

Requirements (1), (2), and (4) clearly boil down to intuitions we may have about set theory in general. We expect that fuzzy set intersection, just like crisp set intersection, is commutative and associative, and that the intersection of a set with its universe yields the original set. Requirement (3) is less straightforward, but still boils down to a simple intuition about fuzzy concepts: We pick a certain element from a fuzzy set A, and discover it has membership degree z. Then we pick the same element from B, and C and find it is a member in these sets to degrees x and y respectively. Further suppose that $x \leq y$, i.e. the element is more of a member in B, than in C. If we now take the intersections $B \cap_T A$ and $C \cap_T A$, we expect that the element will also be more of a member in $B \cap_T A$ than in $C \cap_T A$. Putting it even more nonchalant, we could say that non-decreasingness requires a conjunction to be at most "as true" as its "falsest" conjunct. To show that our definition is again perfectly consistent with crisp set theory let us formulate our intuitive expectations about α-cuts of fuzzy intersections, and crisp intersections of α-cuts.

Consider two fuzzy concepts A and B and an arbitrary way to strictly evaluate their membership criteria. That is to say, fix fuzzy sets A and B, and an α_1. Now the crisp sets $A^{>\alpha_1}$ and $B^{>\alpha_1}$ approximate the fuzzy concepts A and B respectively at degree of strictness α_1. The classical set intersection of these crisp sets yields another crisp set $A^{>\alpha_1} \cap B^{>\alpha_1}$. By intuition we would expect that this crisp set approximates the fuzzy

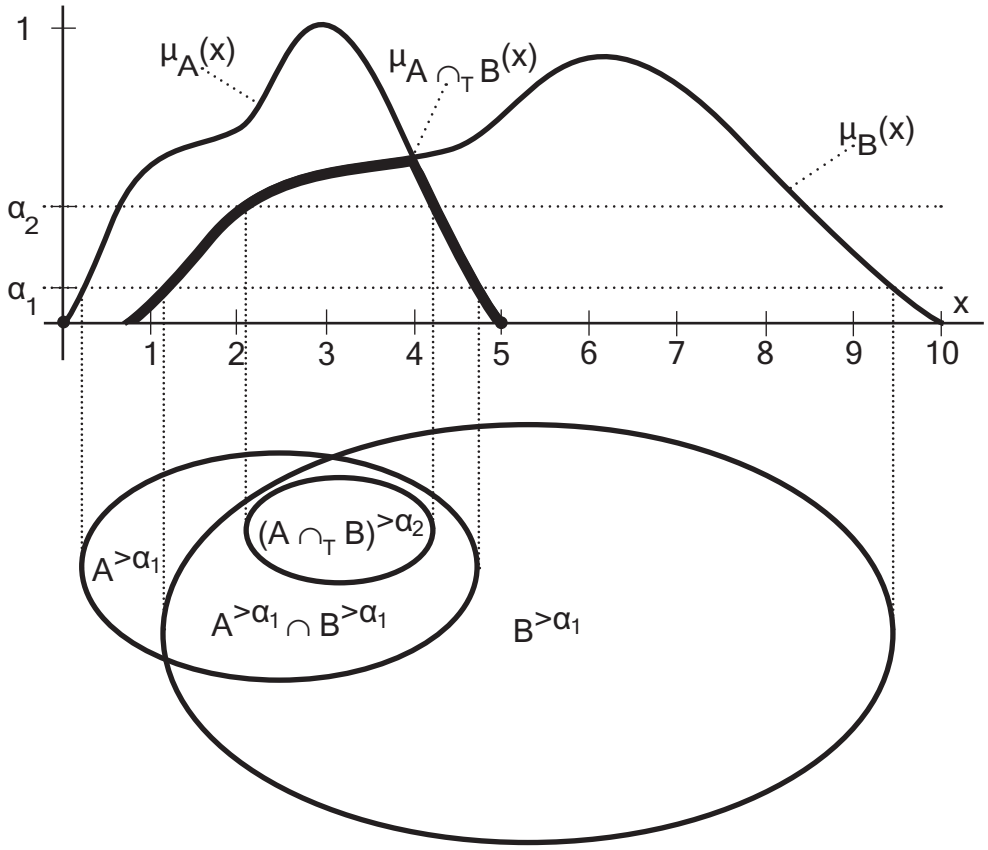


Figure 7: The alpha cut of the T-intersection of fuzzy sets.

concept $A \cap_T B$, again at a degree of strictness α_1. Now consider the (intuitive) fuzzy concept $A \cap_T B$ and think of a way to evaluate its membership criteria at least as strictly as those of A or B in isolation. That is to say, fix a fuzzy set-intersection of A and B and call it $A \cap_T B$, and fix an $\alpha_2 \geq \alpha_1$. Now the crisp set $(A \cap_T B)^{>\alpha_2}$ approximates the fuzzy concept $(A \cap_T B)$ at a degree of strictness which is at least α_1. Thus a variant of Theorem 2 would have to hold:

Lemma 3. *For any fuzzy sets A and B it is the case that for all α_1 and α_2*

$$\alpha_1 \leq \alpha_2 \Rightarrow (A \cap_T B)^{>\alpha_2} \subseteq A^{>\alpha_1} \cap B^{>\alpha_1}.$$

Proof. We will show that for all x in the universe

$$x \in (A \cap_T B)^{>\alpha_2} \Rightarrow x \in (A^{>\alpha_1} \cap B^{>\alpha_1}).$$

For the sake of contradiction, assume that

$$x \in (A \cap_T B)^{>\alpha_2} \quad \wedge \quad x \notin (A^{>\alpha_1} \cap B^{>\alpha_1}).$$

Then, by definitions 6 and 10, we get

$$T(\mu_A(x), \mu_B(x)) > \alpha_2 \quad \wedge \quad (\mu_A(x) \leq \alpha_1 \vee \mu_B(x) \leq \alpha_1).$$

Since, by hypothesis we have $\alpha_2 \geq \alpha_1$, we can rewrite this as

$$T(\mu_A(x), \mu_B(x)) > \mu_A(x) \quad \vee \quad T(\mu_A(x), \mu_B(x)) > \mu_B(x). \tag{5}$$

Recall the requirements of T for commutativity, for the neutral element, and for non-decreasingness from definition 11

$$\forall x, y : T(x, y) = T(y, z).$$

$$\forall x, y : T(x, 1) = x,$$

$$\forall x, y, z : x \leq y \Rightarrow T(x, z) \leq T(y, z).$$

By letting $y = 1$ in the non-decreasingness condition, and substituting from the neutral element and commutativity we get

$$\forall x, z : \quad T(x, z) \leq z \quad \wedge \quad T(x, z) \leq x \tag{6}$$

in contradiction to (5). □

We conclude that our intuition about α-cuts and fuzzy set intersections agrees with the theoretic model introduced so far, in that both predict the same upper bound on the elements in a fuzzy set intersection defined in terms of the well-known intersection operator from classical set theory. This goes well with our intuition, that all elements that are members of the intersection of A and B at least with degree α will also be a member in each of the individual sets A and B at least with degree α.

It is an easy exercise to prove that, for the case of $T_{\min}$ we also get a lower bound of the form $A^{>\alpha_3} \cap B^{>\alpha_3} \subseteq (A \cap_{T_{\min}} B)^{>\alpha_2}$ for $\alpha_2 \leq \alpha_3$. In general, however, we do not require this property of a triangular norm. This is why our converse intuition that all elements that are members in each of A and B at least with degree α will also be a member of the intersection at least with degree α fails, unless we choose $T_{\min}$ as a triangular norm.

6.4 Fuzzy relations and their images

Recall that an n-ary crisp relation $R \in \mathcal{P}(X_1 \times X_2 \times \cdots \times X_n)$ is a subset of the carthesian product $X_1 \times X_2 \times \ldots \times X_n$, i.e. a set of tupes of the form $(x_1, x_2, \ldots, x_n)$ for $x_1 \in X_1, x_2 \in X_2, \ldots, x_n \in X_n$. If a particular tuple $(x_1, x_2, \ldots, x_n) \in R$, we say that x_1, x_2, …, x_n are R-related.

Definition 12. R is an n-ary crisp relation on $X_1 \times X_2 \times \cdots \times X_n$ iff $R \in \mathcal{P}(X_1 \times X_2 \times \ldots \times X_n)$.

In Fuzzy Logic, instead of talking about whether or not x_1, x_2, …, x_n are R-related, we will now talk about degrees to which these are R-related. Therefore the following definition is a straightforward generalization of the crisp case:

Definition 13. We call R an n-ary fuzzy relation on $X_1 \times X_2 \times \cdots \times X_n$ iff $R \in \mathcal{F}(X_1 \times X_2 \times \ldots \times X_n)$.

If a particular tuple $\mu_R(x_1, x_2, \ldots, x_n) = d$ has membership degree d in R, we say that x_1, x_2, …, x_n are R-related to degree d.

Definition 14. Recall that if R is an n-ary crisp relation on $X_1 \times X_2 \times \cdots \times X_n$, and A is a crisp subset of X_1, then the image of A with respect to R, denoted $R(A)$ is

$$R(A) = \{(x_2, x_3, \ldots, x_n) \in X_2 \times X_3 \times \cdots \times X_n \mid \exists x_1 \in X_1 : x_1 \in A \wedge (x_1, x_2, \ldots, x_n) \in X_1 \times X_2 \times \cdots \times X_n\}$$

We could also express this in terms of its (crisp) characteristic function:

$$\chi_{R(A)}(x_2, \ldots, x_n) = \begin{cases} 1, & \text{if } \exists x_1 \in X_1 : x_1 \in A \wedge (x_1, \ldots, x_n) \in X_1 \times \cdots \times X_n \\ 0, & \text{otherwise} \end{cases}.$$

Intuitively, the image $R(A)$ of A with respect to an n-ary relation R reduces the arity of R by essentially binding one (in our definition the first) element in the n-tuple by an existential quantifier.[1] Thus, $(x_2, \ldots, x_n) \in R(A)$ iff $x_2, \ldots, x_n$ are R-related to some $x_1 \in A$. Given that, all we need to do, to define images of fuzzy relations is to translate the crisp proposition $\exists x_1 \in X_1 : x_1 \in A \wedge (x_1, \ldots, x_n) \in X_1 \times \cdots \times X_n$ into the fuzzy domain.

Definition 15. If R is an n-ary fuzzy relation on $X_1 \times X_2 \times \cdots \times X_n$, and A is a fuzzy subset of X_1, then the image of A with respect to R, denoted $R(A)$ is given by the characteristic function

$$\mu_{R(A)}(x_2, x_3, \ldots, x_n) = \sup_{x_1 \in X_1} \{T(\mu_A(x), \mu_R(x_1, x_2, \ldots, x_n)\}$$

for a particular choice T of a triangular norm.

[1] Computational Linguists will note the resemblance of this with the application of an entity to a lambda expression.

We used a triangular norm T in place of the crisp conjunction and the supremum as an aggregation-function over all $x_1 \in X_1$ in place of the existential quantifier. In the previous section we have explained some of the intuition behind the use of triangular norms to capture propositional conjunction. The choice of the supremum as an operation to capture our intuition of existential quantification is less obvious. Note, in this context, that for the discrete case, for instance, where X_1 is an enumerable set, the supremum coincides with a maximum, and the existential quantifier coincides with a disjunction. The maximum happens to be a triangular conorm, the kind of function that resembles disjunction by common fuzzy logic wisdom. It is an easy exercise to see that, if $\min(x, y)$ resembles $x \wedge y$ and $1 - x$ resembles $\neg x$, then $\max(x, y)$ resembles $x \vee y$, by proving that DeMorgan's laws hold in both domains.

Further notation

For the rest of this paper we will let expressions of the form $A(x)$ denote, the value $\mu_A(x)$ of the characteristic function of a fuzzy set A for element x, and expressions of the form $A(x_1, x_2, \ldots, x_n)$ denote the value $\mu_A(x_1, x_2, \ldots, x_n)$ of the characteristic function of an n-ary fuzzy relation A for the tuple $(x_1, x_2, \ldots, x_n)$.

7 The linguistic tool: Syntax-driven semantic analysis

Syntax-driven semantic analysis is the predominant approach to semantic analysis, widely accepted throughout the communities of computational linguistics and compiler construction. Its fundamentals can be traced back to Montague (1973) in the domain of natural languages, and Knuth (1968) in the domain of programming languages. Following this approach, any grammatical production rule p is viewed to serve two functions: a syntactic, and a semantic one.

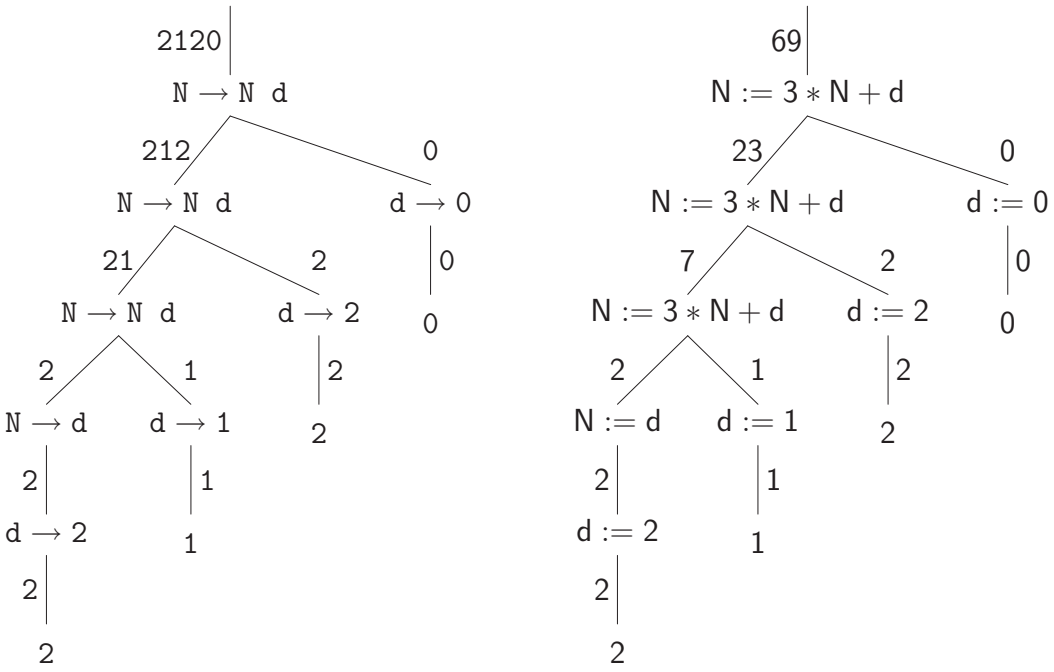
7.1 An example for syntax-driven semantic analysis

Consider a very well-known problem of semantic analysis: the conversion of a number in textual representation to its numeric representation. For example say we were trying to characterize the numeric values of text that represents a number in base-3 notation. Our ultimate goal would be to assign any string $\mathtt{s} \in \{\mathtt{0}, \mathtt{1}, \mathtt{2}\}^*$ (i.e. the syntax of a base-3-number) to its value $\mathtt{s} \in \mathbb{N}$ (i.e. the semantics of a base-3-number). We can characterize the syntax of base-3-numbers by a context-free grammar as shown in the *syntax* column of Figure 8. The table assigns to each syntactic rule a semantic rule.

Figure 9 shows two trees. One depicts the hierachical composition of the syntax of a text from its syntactic parts, the other depicts the hierachical composition of the semantics of the same text from the semantics of its parts. It can be seen that both trees share the same structure. This is a very convenient feature, because the construction of parse trees is a well understood topic. Once we have established such a one-to-one correspondence between a context-free production rule and its inherent semantics we can

syntax	semantics
$\mathtt{d} \to 0$	$\mathsf{d} := 0$
$\mathtt{d} \to 1$	$\mathsf{d} := 1$
$\mathtt{d} \to 2$	$\mathsf{d} := 2$
$\mathtt{N} \to \mathtt{d}$	$\mathsf{N} := \mathsf{d}$
$\mathtt{N} \to \mathtt{N}\ \mathtt{d}$	$\mathsf{N} := 3 * \mathsf{N} + \mathsf{d}$

Figure 8: A syntactic grammar for base-3-numbers and the semantic function of each rule.



(a) syntactic derivation

(b) semantic derivation

Figure 9: The analysis of a text 2120 for its numeric value 2120_3.

build a representation for the meaning of a text as a "side effect" to the syntactic parse, by simply applying a semantic function to nodes in the tree as we apply a syntactic reduction.

In analogy to our model for the conversion of base-3 numbers given in Figure 8, Figure 10 shows a model for the analysis of English text with respect to our relational model. The central observation that allowed us to characterize base-3 numbers by means of syntax-driven semantic analysis was the fact, that we can semantically derive the meaning of a string that is syntactically derived by $\mathtt{N} \rightarrow \mathtt{N\ d}$ as $\mathsf{N} := 3 * \mathsf{N} + \mathsf{d}$. For the sample grammar in Figure 10, it is the concept of the image of a fuzzy relation that "does the trick".

7.2 Semantic context-free grammars

Definition 16. A semantic context-free grammar $G = (\mathtt{V}, \mathtt{T}, P, \mathtt{S}, \mathsf{Dom})$ consists of

- A finite non-empty set $\mathtt{V}$ of *non-terminal* symbols.
- A finite non-empty set $\mathtt{T}$ of *terminal* symbols.
- A set P of production rules, each of which is of the form $(\mathtt{A} \rightarrow \mathtt{a_1 a_2 \ldots a_n}, \mathsf{eval})$ where $\mathtt{A} \in \mathtt{V}$ is a grammar variable, and each $\mathtt{a}_i$ is a symbol, either terminal or non-terminal, i.e. $\mathtt{a}_i \in (\mathtt{V} \cup \mathtt{T})$. Let u be the number of symbols in $\mathtt{a_1 a_2 \ldots a_n}$ that are non-terminal, so that $n - u$ is the number of symbols that are terminal. Then $\mathsf{eval} : \mathsf{Dom}^u \mapsto \mathsf{Dom}$ is a mapping from Dom^u to Dom. In the special case where $u = 0$, eval is taken to be a constant value $\mathsf{eval} \in \mathsf{Dom}$.
- A *start-symbol* $\mathtt{S} \in \mathtt{V}$.
- A finite non-empty set Dom establishing the semantic *domain*.

Definition 17. Fix a semantic context-free grammar $G = (\mathtt{V}, \mathtt{T}, P, \mathtt{S}, \mathsf{Dom})$. We say that a non-terminal symbol $\mathtt{X} \in \mathtt{V}$ assigns a string $\alpha = \mathtt{t_1 t_2 \ldots t_n}$ of terminals $\mathtt{t}_i \in T$ the meaning $\mathsf{X} \in \mathsf{Dom}$ in G iff $(\mathtt{X} \rightarrow \mathtt{x_1 x_2 \ldots x_m}, \mathsf{eval})$ is a production in P where for each $\mathtt{x}_i$

- $\mathtt{x}_i \in \mathtt{V}$ and $\mathtt{x}_i$ assigns the string of terminals α_i the meaning x_i, or
- $\mathtt{x}_i \in T$, in which case we take α_i to be $\mathtt{x}_i$

and α is the concatenation $\alpha_1 \cdot \alpha_2 \cdot \ldots \cdot \alpha_m$ and $\mathsf{X} = \mathsf{eval}(\mathsf{x}'_1, \mathsf{x}'_2, \ldots, \mathsf{x}'_n)$, where $\langle \mathsf{x}'_1, \mathsf{x}'_2, \ldots, \mathsf{x}'_u \rangle$ is the sequence $\langle \mathsf{x}_1, \mathsf{x}_2, \ldots, \mathsf{x}_n \rangle$ with all the elements removed that represent meanings of non-terminals. In the special case where $u = 0$, $\mathsf{eval}()$ is taken to evaluate to a constant value $\mathsf{eval} \in \mathsf{Dom}$, as defined in the production rule.

8 The syntax/semantics interface: making ends meet

Figure 10 gives semantic rules for all syntactic productions of our example grammar to capture their fuzzy semantics with respect to our example data model.

The basic idea is that we can view the meaning of any phrase or word in this grammar as a fuzzy relation, and that we can determine such a fuzzy relation for each possible word or phrase that appears in our language, from fuzzy relations describing its sub-phrases or from data we have in our database.

syntax	semantics
$\texttt{S} \rightarrow \texttt{NP VP}$	$\mathsf{S}(x) := \sup_y\{T\Big(\mathsf{NP}(y), \mathsf{VP}(x,y)\Big)\}$
$\texttt{VP} \rightarrow \texttt{V PP}$	$\mathsf{VP}(x,\lambda_y) := \sup_z\{T\Big(\mathsf{V}(x,\lambda_y,z), \mathsf{PP}(z)\Big)\}$
$\texttt{VP} \rightarrow \texttt{V NP}$	$\mathsf{VP}(x,\lambda_y) := \sup_z\{T\Big(\mathsf{V}(x,\lambda_y,z), \mathsf{NP}(z)\Big)\}$
$\texttt{NP} \rightarrow \texttt{Nom}$	$\mathsf{NP}(x) := \mathsf{Nom}(x)$
$\texttt{NP} \rightarrow \texttt{Det N'}$	$\mathsf{NP}(x) := \mathsf{N'}(x)$
$\texttt{N'} \rightarrow \texttt{N}$	$\mathsf{N'}(x) := \mathsf{N}(x)$
$\texttt{N'} \rightarrow \texttt{AP N}$	$\mathsf{N'}(x) := T\Big(\mathsf{AP}(x), \mathsf{N}(x)\Big)$
$\texttt{N'} \rightarrow \texttt{N' PP}$	$\mathsf{N'}(x) := T\Big(\mathsf{N'}(x), \mathsf{PP}(x)\Big)$
$\texttt{AP} \rightarrow \texttt{Adj}$	$\mathsf{AP}(x) := \mathsf{Adj}(x)$
$\texttt{AP} \rightarrow \texttt{very AP}$	$\mathsf{AP}(x) := \Big(\mathsf{AP}(x)\Big)^2$
$\texttt{PP} \rightarrow \texttt{in NP}$	$\mathsf{PP}(x) := \mathsf{NP}(x)$
$\texttt{PP} \rightarrow \texttt{near NP}$	$\mathsf{PP}(x) := \sup_y\{T\Big(\mathsf{NP}(y), \max(\min(\frac{50\ \text{km}-d}{50\ \text{km}-20\ \text{km}}, 1), 0\Big) \mid \mathsf{Place\_Distance}(x,y,d)\}$
$\texttt{V} \rightarrow \texttt{lives}$	$\mathsf{V}(x,\lambda_y,\lambda_z) := 1.0$ if $\mathsf{Lives\_In}(x,\lambda_y,\lambda_z)$, 0.0 otherwise
$\texttt{V} \rightarrow \texttt{likes}$	$\mathsf{V}(x,\lambda_y,\lambda_z) := 1.0$ if $\mathsf{Likes}(x,\lambda_y,\lambda_z)$, 0.0 otherwise
$\texttt{Adj} \rightarrow \texttt{small}$	$\mathsf{Adj}(x) := \max\Big(\min(\frac{20000-p}{20000-10000}, 1), 0\Big) \mid \mathsf{Place\_Population}(x,p)$
$\texttt{N} \rightarrow \texttt{city}$	$\mathsf{N}(x) := 1.0$ if $\mathsf{Place}(x)$, 0.0 otherwise
$\texttt{Nom} \rightarrow \texttt{Carol}$	$\mathsf{Nom}(x) := 1.0$ if $\mathsf{Person\_Name}(x, \mathsf{carol})$, 0.0 otherwise
$\texttt{Nom} \rightarrow \texttt{Frank}$	$\mathsf{Nom}(x) := 1.0$ if $\mathsf{Person\_Name}(x, \mathsf{frank})$, 0.0 otherwise
$\texttt{Nom} \rightarrow \texttt{San Fr.}$	$\mathsf{Nom}(x) := 1.0$ if $\mathsf{Place\_Name}(x, \mathsf{san\ francisco})$, 0.0 otherwise
$\texttt{Det} \rightarrow \texttt{a}$	$\mathsf{Det}(x) := 0$

Figure 10: Our example semantic grammar

In the following section we will go through the most important rules of the resulting semantic grammar. Applying the technique of syntax-driven semantic analysis to this grammar, we can then derive fuzzy sets resembling the meanings of any grammatically correct expression.

8.1 The semantics of nominals

Our definition of the meanings of nominals relies on our database containing data about the names of entities we wish to refer to by nominals. The relation Person_Name, for example, associates a name to a person.

`Nom` → `Carol`	$\mathsf{Nom}(x) := 1.0$ if $\mathsf{Person\_Name}(x, \mathsf{carol})$, 0.0 otherwise
`Nom` → `Frank`	$\mathsf{Nom}(x) := 1.0$ if $\mathsf{Person\_Name}(x, \mathsf{frank})$, 0.0 otherwise
`Nom` → `San Fr.`	$\mathsf{Nom}(x) := 1.0$ if $\mathsf{Place\_Name}(x, \mathsf{san\ francisco})$, 0.0 otherwise

We can describe the meaning of any nominal by means of a set of things x, such that the $\mathsf{Nom}(x)$ condition is fulfilled. Recall that $\mathsf{Nom}(x)$ is the characteristic function of a fuzzy set (unary fuzzy relation), in this case, of all things x in our domain that are called `Carol`, `Frank`, or `San Francisco`.

Note that, in our first naive approach, we have assigned crisp meanings to all nominals. Therefore, in our grammar, it makes sense to state that someone's name is *Carol*, but it does not make sense to state that someone's name is *Carol* to a degree of 0.7.

8.2 The semantics of nouns

As opposed to nominals, which pick out specific things from the domain by referring to them by name, nouns can be abstractions. For example the set of all cities is given by

`N` → `city` $\quad \mathsf{N}(x) := 1.0$ if $\mathsf{Place}(x)$, 0.0 otherwise

in our example grammar. Here we chose the simplistic approach of, again, constructing a crisp set of all places in our database, to represent the set of all things referred to as `city`.

In a real-world example, this might, however, already be a candidate for a fuzzy concept, since one might want to employ certain criteria to decide whether something is a city, possibly based on measures of population or area to distinguish a *city* from, say, a *town* or a *metropolis*.

8.3 The semantics of adjectives

As our simple example grammar follows the approach of intersective semantics the meaning of an adjective like `small` will be the set of all small things, in much the same way as the meaning of the noun phrase `small thing` would be the set of all small things.

`Adj` → `small` $\quad \mathsf{Adj}(x) := \max\left(\min(\frac{20000-p}{20000-10000}, 1), 0\right) \mid \mathsf{Place\_Population}(x, p)$

Note that the adjective `small` is a perfect example for a fuzzy concept. What does it mean, as in this particular case for a city, to be `small`? A fuzzy set could easily be set up that takes into account several measures of "smallness" such as population, area

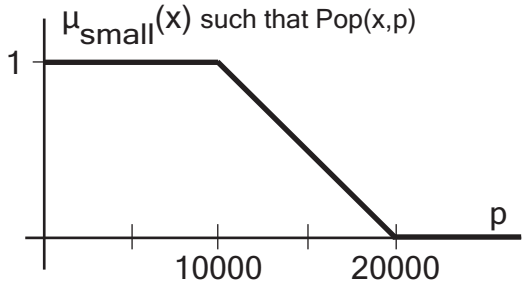


Figure 11: The characteristic function of `small`.

or density of infrastructure, etc. In our simple approach we only use population as a measurement of the size of a city and define a fuzzy decision boundary on this measure to determine whether a place is a small city.

More particularly, let $\mathsf{Place\_Population}(x, p)$. Then the above definition assigns a degree of fulfillment $\mathsf{Adj}(x) = 1.0$ to all cities x whose population $p \leq 10000$, a degree of fulfillment 0.0 to all cities whose population $p \geq 20000$, and a degree of fulfillment $\frac{20000-p}{20000-10000}$ which amounts to a linear interpolation between these two points to all cities whose population is between 10000 and 20000.

8.4 The semantics of adjectival phrases

In our sample grammar, adjectival phrases can consist of nothing but an adjective. In this case it is quite straightforward to assume that the meaning of the adjectiveal phrase will be exactly the same as for the only adjective it contains.

$\texttt{AP} \rightarrow \texttt{Adj}$	$\mathsf{AP}(x) := \mathsf{Adj}(x)$
$\texttt{AP} \rightarrow \texttt{very AP}$	$\mathsf{AP}(x) := \Big(\mathsf{AP}(x)\Big)^2$

To exemplify the case where the phrase consists of an adverb followed by an adjectival phrase, we chose *very* which serves as a classic example in the fuzzy logic literature of what it calls a linguistic hedge.

One of the most simplistic techniques to construct the meaning for a fuzzy set resembling `veryAP` from the meaning of a fuzzy set resembling `AP` is to square the degrees of fulfillment. We stick with this solution once proposed by Zadeh for its simplicity. However, the reader should note that a much more adquate treatment of the famous hedges is given by De Cock & Kerre (2002).

8.5 The semantics of prepositional phrases

Again, in consequence to our usage of intersective semantics, the meaning of a prepositional phrase like `in San Francisco` will be the set of all things that are `in San Francisco`, and thus the same as the noun phrases `thing in San Francisco` or `San Francisco` on its own.

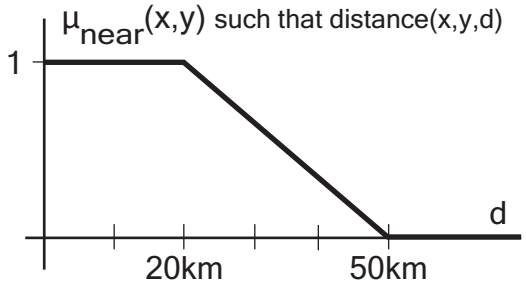


Figure 12: The characteristic function of `near`.

$$
\begin{array}{ll}
\texttt{PP} \rightarrow \texttt{in NP} & \mathsf{PP}(x) := \mathsf{NP}(x) \\
\texttt{PP} \rightarrow \texttt{near NP} & \mathsf{PP}(x) := \sup_y \{ \\
& \quad T\Big(\mathsf{NP}(y), \max(\min(\tfrac{50\text{ km}-d}{50\text{ km}-20\text{ km}}, 1), 0)\Big) \\
& \quad \mid \mathsf{Place\_Distance}(x, y, d)\}
\end{array}
$$

Things get more interesting considering the preposition `near`, another example for a true fuzzy concept. The approach we chose was to use geographic distance as a measurement of how close two cities are to each other, and to define a fuzzy decision boundary on this measure, to determine, given two cities `x` and `y` to which degree `x` satisfies the constraint that it should be `near y`.

More particularly, let $\mathsf{Place\_Distance}(x, y, d)$. Then the above definition assigns a degree of fulfillment $\mathsf{PP}(x) = 1.0$ to all cities x whose geographic distance d from y satisfies $d \leq 20$ km, a degree of fulfillment 0.0 to all cities with $d \geq 50$ km, and a degree of fulfillment $\frac{50\text{ km}-d}{50\text{ km}-20\text{ km}}$ which amounts to a linear interpolation between these two points to all cities with $20\text{ km} < d < 50$ km.

8.6 The semantics of noun phrases

For our approach of intersective semantics, the definition of the meaning of noun-phrases turns out to be quite straightforward. Our grammar allows for noun phrases that either rewrite to a nominal (in which case the meaning of the nominal is simply preserved in the noun phrase) or to a determiner followed by a category we called `N'`. Here we will take the simplistic approach that a determiner doesn't contribute to the meaning of a noun phrase, and pass up the meaning of the `N'` to the noun phrase. An `N'`, in turn, can rewrite to a noun (in which case, again, we simply preserve the noun's meaning and pass it up to the noun phrase), or be premodified by an adjectival phrase or postmodified by a prepositional phrase.

$$
\begin{array}{ll}
\texttt{NP} \rightarrow \texttt{Nom} & \mathsf{NP}(x) := \mathsf{Nom}(x) \\
\texttt{NP} \rightarrow \texttt{Det N'} & \mathsf{NP}(x) := \mathsf{N'}(x) \\
\texttt{N'} \rightarrow \texttt{N} & \mathsf{N'}(x) := \mathsf{N}(x) \\
\texttt{N'} \rightarrow \texttt{AP N} & \mathsf{N'}(x) := T\Big(\mathsf{AP}(x), \mathsf{N}(x)\Big) \\
\texttt{N'} \rightarrow \texttt{N' PP} & \mathsf{N'}(x) := T\Big(\mathsf{N'}(x), \mathsf{PP}(x)\Big)
\end{array}
$$

As we've explained before, intersective semantics treats any kind of modification as an

intersection. Therefore the meaning of a noun phrase like `very small city` is simply the set AP of all things that are `very small` intersected with the set N of all things that are cities, and the meaning of a noun phrase like `a city near San Francisco` is simply the set of all things that are a `city` and the set of all things that are `near San Francisco`.

We've explained before why we can use triangular norms to capture intersections of fuzzy constraints in a straightforward and intuitive way, and this is exactly what these rules make use of.

8.7 The semantics of verbs

So far we have only dealt with, what we where casually referring to as "things". We have represented the semantics of phrases headed by nouns, adjectives and prepositions by means of fuzzy sets that select certain objects from our domain.

In order to capture the semantics of verbs, we will need to make use of fuzzy relations. For example the meaning of `likes` in our grammar, would be the relation $\mathsf{V}(x, y, z)$ that holds between three elements from our domain x, y, z if, and only if, x is an eventuality involving some person y liking some person z.

$\texttt{V} \rightarrow \texttt{lives}$ $\quad \mathsf{V}(x, \lambda_y, \lambda_z) := 1.0$ if $\mathsf{Lives\_In}(x, \lambda_y, \lambda_z)$, 0.0 otherwise
$\texttt{V} \rightarrow \texttt{likes}$ $\quad \mathsf{V}(x, \lambda_y, \lambda_z) := 1.0$ if $\mathsf{Likes}(x, \lambda_y, \lambda_z)$, 0.0 otherwise

Here we have, again, assumed that the eventuality that these verbs denote are not actually fuzzy concepts. One can, however, imagine situations in which verbs refer to fuzzy concepts. For example if we had a data model that contains facts about moving people, we might use the velocity with which people are moving to determine whether someone `strolls`, `walks`, or `runs`.

8.8 The semantics of verb phrases

Now that we know what it means for some factoid x to refer to the action of some person λ_y to `like` some other person λ_z (i.e. $\mathsf{V}(x, \lambda_y, \lambda_z)$), what does it mean for x to refer to the action of λ_y to `like Carol`? The way we go about this is to state that x refers to the action of λ_y to `like Carol` if, and only if, *there exists* some person z such that x refers to the action of λ_y liking z (i.e. $\mathsf{V}(x, \lambda_y, z)$) *and* z refers to something described by `Carol` (i.e. $\mathsf{NP}(z)$).

$\texttt{VP} \rightarrow \texttt{V PP}$ $\quad \mathsf{VP}(x, \lambda_y) := \sup_z\{T\Big(\mathsf{V}(x, \lambda_y, z), \mathsf{PP}(z)\Big)\}$
$\texttt{VP} \rightarrow \texttt{V NP}$ $\quad \mathsf{VP}(x, \lambda_y) := \sup_z\{T\Big(\mathsf{V}(x, \lambda_y, z), \mathsf{NP}(z)\Big)\}$

As mentioned earlier, we can translate an existential quantifier to the fuzzy domain, by means of a supremum, and the conjunction by means of a triangular norm. Thus the meaning VP of a `VP` given the meaning V of a `V` and the meaning NP of a `NP` will amount to the fuzzy image $\mathsf{V}(\mathsf{NP})$ of NP with respect to V.[2] The meaning of verb phrases consisting of a verb and a prepositional phrase, can be defined by $\mathsf{V}(\mathsf{PP})$ in analogy.

[2] Again, readers accustomed to working with lambda calculus will find the expression $\mathsf{V}(\mathsf{NP})$ to capture the meaning of a verb phrase quite familiar.

8.9 The semantics of sentences

Now that we know what it means for some factoid x to refer to the action of some person λ_y to `like Carol` (i.e. $\mathsf{VP}(x, \lambda_y)$), what does it mean for x to refer to the action described by `Frank likes Carol`? In analogy to our definition of the semantics of verb phrases, we go about this by stating that x refers to the action described by `Frank likes Carol` if, and only if, *there exists* some person y such that x refers to the action of y liking Carol (i.e. $\mathsf{VP}(x, y)$) *and* y refers to something described by `Frank` (i.e. $\mathsf{NP}(y)$).

$$\texttt{S} \to \texttt{NP}\ \ \texttt{VP} \quad \mathsf{S}(x) := \sup_y\{T\Big(\mathsf{NP}(y), \mathsf{VP}(x, y)\Big)\}$$

Thus the meaning S of an `S` given the meaning NP of a `NP` and the meaning VP of a `VP` will amount to the fuzzy image $\mathsf{VP}(\mathsf{NP})$ of NP with respect to VP.

8.10 Fuzzy relational semantics of context-free languages

Definition 18. A context-free grammar with fuzzy relational semantics $G = (\texttt{V}, \texttt{T}, P, \texttt{S}, \mathsf{SDom}, \mathsf{Rel})$ is defined by

- a semantic context-free grammar $(\texttt{V}, \texttt{T}, P, \texttt{S}, \mathit{GDom})$, and
- a relational scheme $(\mathit{SDom}, \mathit{Rel})$

where $\mathit{GDom} = \bigcup_i \mathcal{F}(\mathit{SDom}^i)$ over all i between zero and the maximal number of non-terminals that appear in the body of any production rule in P.

Definition 19. Fix a context-free grammar with fuzzy relational semantics $G = (\texttt{V}, \texttt{T}, P, \texttt{S}, \mathsf{SDom}, \mathsf{Rel})$. We say that `X` assigns a string α the meaning X in G iff `X` assigns α the meaning X in $(\texttt{V}, \texttt{T}, P, \texttt{S}, \mathit{GDom})$ with GDom defined by SDom as above.

Definition 20. Fix a context-free grammar with fuzzy relational semantics $G = (\texttt{V}, \texttt{T}, P, \texttt{S}, \mathsf{SDom}, \mathsf{Rel})$. The fuzzy semantics of a string $\alpha = \texttt{t}_1\texttt{t}_2 \ldots \texttt{t}_\texttt{n}$ of terminals $\texttt{t}_i \in T$ with respect to G is given by the binary relation $\mu_\mathsf{S}(x) \leq \mu_\mathsf{S}(y)$ on $\mathsf{SDom} \times \mathsf{SDom}$ where `S` assigns α the meaning S in G.

Note that, since the relation $\leq$ on the unit-interval is a weak ordering, so is the relation $\mu_\mathsf{S}(x) \leq \mu_\mathsf{S}(y)$ on $\mathsf{SDom} \times \mathsf{SDom}$.

9 Concluding remarks and future directions

In this paper a grammatical framework was shown that augments context-free production rules with semantic production rules that rely on fuzzy relations as representations of fuzzy natural language concepts. Furthermore, it was shown how the well-known technique of syntax-driven semantic analysis can be used to infer from such a semantically augmented grammar the semantics of a given expression, where the semantics of expressions are taken to be orderings on the possible worlds they describe.

More specifically we were considering the application of natural language query processing to motivate our studies. We assumed that we were given a relational scheme on a certain domain, and showed how we could arrive at an ordering of that domain according

to the degree to which its elements satisfy the constraint specified by means of a natural language statement. We considered the specific example of a relational database on the domain of people and places containing information about which cities people live in, populations of cities, and distances between cities, and showed how exactly our technique could be applied to arrive at an ordered sequence of records satisfying the natural language query statement `Carol lives in a small city near San Francisco`.

The primary aim of this paper was to demonstrate the overall approach by particular examples and to make a first attempt at defining what exactly a fuzzy semantic grammar may look like, and how it is to be interpreted. However, it raises a number of questions that were not covered herein. Most importantly: Will the approach scale to cover the full complexity of the semantic microdomains that are of interest today, such as typical industrial or scientific knowledge bases, and the full expressive power of natural languages?

On the semantic side this clearly raises questions about computational complexity and about how inference mechanisms can best be incorporated into the system. On the syntactic side this certainly requires more sophisticated linguistic formalisms to be investigated in the context of fuzzy semantics than plain context-free grammars. It seems promising to define the fuzzy semantics of a natural language proposition in terms of a tectogrammatical analysis derived from a system that is more state-of-the-art than the toy-grammar we have used.

Considering the work that lies ahead, we can perhaps only claim to have contributed a small first step towards the actual inception of a precisiated natural language as a basis of a computational theory of perceptions, and a tool that may turn out a useful means to deal with fuzzy concepts in the context of natural language processing applications some day. However, we also made clear that, in the light of modern applications, the pursuit of this line of research probably is more rewarding today than ever before.

Acknowledgments

The author would like to thank Ulrich Bodenhofer for his continued commitment and support of the present work. Without his supervision this work would certainly not have been possible. Furthermore he thanks Ann Copestake, Ted Briscoe and Daniel Osherson for reading early drafts of this report.

References

Bodenhofer, U. & Bauer, P. (2003), A formal model of interpretability of linguistic variables, *in* J. Casillas, O. Cordón, F. Herrera & L. Magdalena, eds, 'Interpretability Issues in Fuzzy Modeling', Vol. 128 of *Studies in Fuzziness and Soft Computing*, Springer-Verlag, Heidelberg, pp. 524–545.

De Cock, M. & Kerre, E. E. (2002), 'A context-based approach to linguistic hedges', *International Journal of Applied Mathematical Computer Science* **12**(3), 371–382.

De Cock, M., Bodenhofer, U. & Kerre, E. E. (2000), Modelling linguistic expressions using fuzzy relations, *in* 'Proceedings of the 6th International Conference on Soft Computing', Iizuka, pp. 353–360.

Dvorak, A. & Novak, V. (2000), On the extraction of linguistic knowledge in databases using fuzzy logic, *in* H. L. Larsen, ed., 'Flexible Query Answering Systems. Recent Advances.', Physica-Verlag, pp. 445–454.

Gaines, B. R. & Kohout, L. J. (1977), 'The fuzzy decade: A bibliography of fuzzy systems and closely related topics', *International Journal of Man-Machine Studies* **9**, 1–68.

Goguen, J. A. (1974), 'Concept representation in natural and artificial languages: Axioms, extensions and applications for fuzzy sets', *International Journal of Man-Machine Studies* **6**, 513–561.

Goguen, J. A. (1975), On fuzzy robot planning, *in* L. A. Zadeh, ed., 'Fuzzy Sets and their Applications to Cognitive and Decision Processes', Academic Press.

Klement, E. P., Mesiar, R. & Pap, E. (2000), *Triangular Norms*, Vol. 8 of *Trends in Logic*, Kluwer Academic Publishers, Dordrecht.

Knuth, D. E. (1968), 'Semantics of context-free languages', *Mathematical Systems Theory* **2**(2), 127–145.

Lee, E. T. & Zadeh, L. A. (1969), 'Note on fuzzy languages', *Information Sciences* **1**, 421–434.

Montague, R. M. (1973), The proper treatment of quantification in ordinary english, *in* K. J. J. Hintikka & J. M. E. Moravcsik, eds, 'Approaches to Natural Language'.

Novak, V. (1991), 'Fuzzy logic, fuzzy sets, and natural languages', *International Journal on General Systems* **20**, 83–97.

Novak, V. (1992), *The Alternative Mathematical Model of Linguistic Semantics and Pragmatics*, Vol. 8 of *IFSR International Series on Systems Science and Engineering*, International Federation for Systems Research.

Novak, V. (1995), 'Linguistically oriented fuzzy logic controller and its design', *International Journal of Approximate Reasoning*.

Novak, V. (1997), Evaluating linguistic expressions and their role in the design of the fuzzy control strategy, *in* N. Steele, ed., 'Proceedings of the Second International ICSC Symposium on Fuzzy Logic and Applications (ISFL '97)', ICSC, ICSC Academic Press, pp. 89–94.

Parret, H. (1974), *Discussing Language*, Vol. 93 of *Janua Linguarum Series Maior*, Mouton.

Sheppard, D. (1954), 'The adequacy of everyday quantitative expressions as measurements of qualities', *British Journal of Psychology* **45**, 40–50.

Zadeh, L. A. (1965), 'Fuzzy sets', *Information and Control* **8**, 338–353.

Zadeh, L. A. (1975*a*), 'The concept of a linguistic variable and its application to approximate reasoning Part I', *Information Sciences* **8**, 199–250.

Zadeh, L. A. (1975*b*), 'The concept of a linguistic variable and its application to approximate reasoning Part II', *Information Sciences* **8**, 301–357.

Zadeh, L. A. (1975*c*), 'The concept of a linguistic variable and its application to approximate reasoning Part III', *Information Sciences* **9**, 43–80.

Zadeh, L. A. (1978), 'Pruf – a meaning representation language for natural languages', *International Journal of Man-Machine Studies* **10**, 395–460.

Zadeh, L. A. (1981), Test-score semantics for natural languages and meaning representation via pruf, *in* B. B. Rieger, ed., 'Empirical Semantics: A collection of new approaches in the field.', Studienverlag Brockmeyer, pp. 281–349.

Zadeh, L. A. (1982), Test-score semantics for natural languages, *in* 'Proceedings of the 9th conference on Computational linguistics', Academia Praha, Czechoslovakia, pp. 425–430.

Zadeh, L. A. (1999), 'From computing with numbers to computing with words. from manipulation of measurements to manipulation of perceptions', *IEEE Trans. Circuits Syst. I* **46**, 105–119.

Zadeh, L. A. (2001), 'A new direction in ai: Toward a computational theory of perceptions', *AI Magazine* **22**, 73–84.

Zadeh, L. A. (2003), From search engines to question-answering systems – the need for new tools, *in* 'Web Intelligence, First International Atlantic Web Intelligence Conference (AWIC 2003)', Vol. 2663 of *Lecture Notes in Computer Science*, Springer, pp. 15–17.

Zadeh, L. A. (2004*a*), 'A note on web intelligence, world knowledge, and fuzzy logic', *Data & Knowledge Engineering* **50**, 291–304.

Zadeh, L. A. (2004*b*), 'Precisiated natural language', *AI Magazine*.

A The Carol example

In order to demonstrate the kind of semantic analysis we have in mind more clearly, let's turn back to the example data presented in Section 4.1, and try to determine a degree to which each of them satisfies the natural language proposition `Carol lives in a small city near San Francisco` with respect to the grammar from Section 5.1 and the semantic mapping from Section 8.

Figure 13 shows the structure of both the syntactic and semantic derivations of this sentence from their respective atomic parts. The nodes in the tree that we can assign a meaning in our framework are assigned labels $N_1, N_2, \ldots, N_{16}$. To each label we will assign a fuzzy relation as a representation of its semantics. Assume, for the sake of simplicity, that we have already carried out a syntactic parse of the sentence, constructed a representation of the parse tree, and are now traversing the nodes of the tree in a in a depth-first post-processing order to assign to each node a semantic representation. During that traversal we assign to each node the semantic representation as a fuzzy relation, which

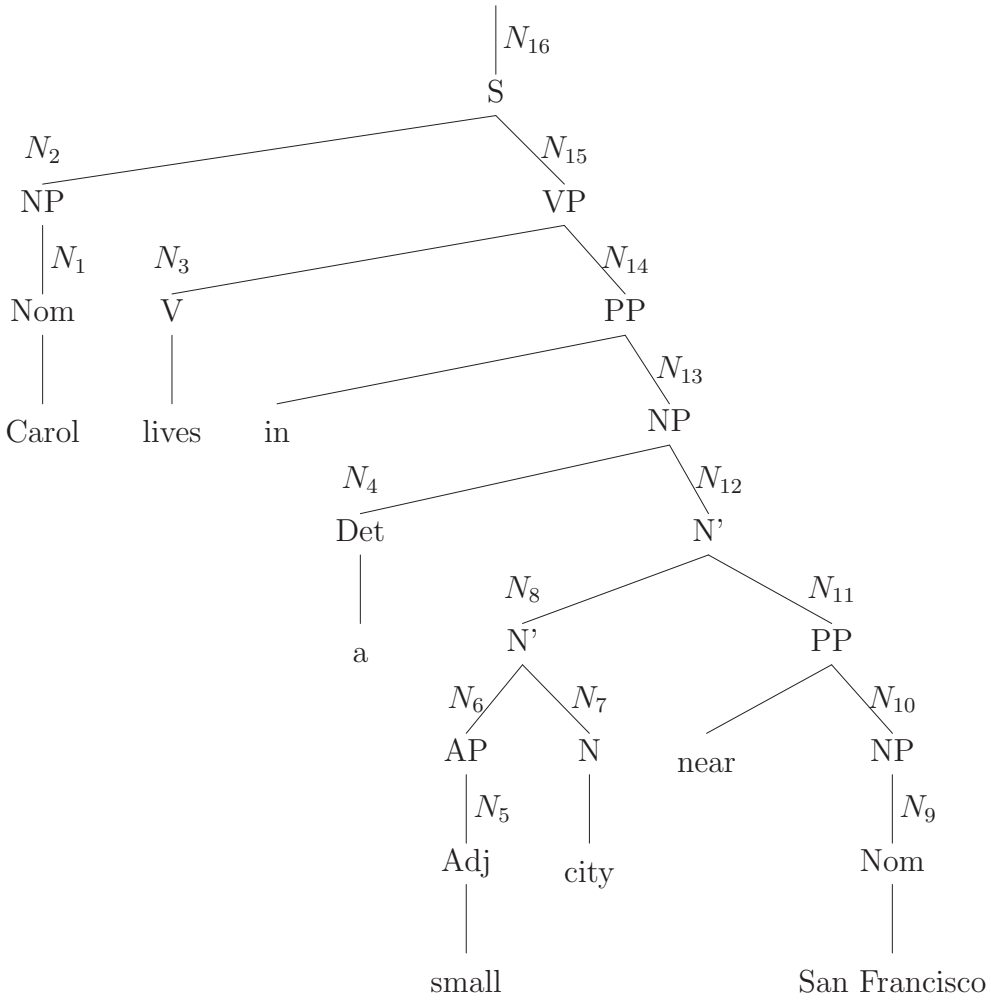


Figure 13: Derivation tree for "`Carol lives in a small city near San Francisco`"

we will represent here by means of crisp set of (*member*, *degree of membership*)-tuples. The triangular norm that will be used throughout the example is the minimum.

N_1 $\mathsf{Nom}(x) = 1.0$ if $\mathsf{Person\_Name}(x, \mathsf{carol})$, 0.0 othw.

N_3 $\mathsf{V}(x, \lambda_y, \lambda_z) = 1.0$ if $\mathsf{Lives\_In}(x, \lambda_y, \lambda_z)$, 0.0 othw.

N_5 $\mathsf{Adj}(x) = \max\big(\min(\frac{20000-p}{20000-10000}, 1), 0\big)$
| $\mathsf{Place\_Population}(x, p)$

N_7 $\mathsf{N}(x) = 1.0$ if $\mathsf{Place}(x)$, 0.0 othw.

N_8 $\mathsf{N'}(x) = T(\ \max\big(\min(\frac{20000-p}{20000-10000}, 1), 0\big),$
1.0 if $\mathsf{Place}(x)$, 0.0 othw. $\ )$
| $\mathsf{Place\_Population}(x, p)$

N_9 $\mathsf{Nom}(x) = 1.0$ if $\mathsf{Place\_Name}(x, \mathsf{san\ francisco})$, 0.0 othw.

N_{11} $\mathsf{PP}(x) = \sup_a\{T(\ 1.0$ if $\mathsf{Place\_Name}(a, \mathsf{san\ francisco})$, 0.0 othw.,
$\max(\min(\frac{50\ \text{km}-d}{50\ \text{km}-20\ \text{km}}, 1), 0)\ )\}$
| $\mathsf{Place\_Distance}(a, y, d)$

N_{12} $\mathsf{N'}(x) = T(\ T(\ \max\big(\min(\frac{20000-p}{20000-10000}, 1), 0\big),$
1.0 if $\mathsf{Place}(x)$, 0.0 othw. $\ ),$
$\sup_a\{T(\ 1.0$ if $\mathsf{Place\_Name}(a, \mathsf{san\ francisco})$, 0.0 othw.,
$\max(\min(\frac{50\ \text{km}-d}{50\ \text{km}-20\ \text{km}}, 1), 0)\ )\}\ )$
| $\mathsf{Place\_Distance}(x, a, d) \wedge \mathsf{Place\_Population}(x, p)$

N_{15} $\mathsf{VP}(x, y) = \sup_z\{T(\ 1.0$ if $\mathsf{Lives\_In}(x, y, z)$, 0.0 othw.,
$T(\ T(\ \max\big(\min(\frac{20000-p}{20000-10000}, 1), 0\big),$
1.0 if $\mathsf{Place}(z)$, 0.0 othw. $\ ),$
$\sup_a\{T(\ 1.0$ if $\mathsf{Place\_Name}(a, \mathsf{san\ francisco})$, 0.0 othw.,
$\max(\min(\frac{50\ \text{km}-d}{50\ \text{km}-20\ \text{km}}, 1), 0)\ )\}\ )\ )\}$
| $\mathsf{Place\_Distance}(z, a, d) \wedge \mathsf{Place\_Population}(z, p)$

N_{15} $\mathsf{S}(x) = \sup_y\{T(\ 1.0$ if $\mathsf{Person\_Name}(y, \mathsf{carol})$, 0.0 othw.,
$\sup_z\{T(\ 1.0$ if $\mathsf{Lives\_In}(x, y, z)$, 0.0 othw.,
$T(\ T(\ \max\big(\min(\frac{20000-p}{20000-10000}, 1), 0\big),$
1.0 if $\mathsf{Place}(z)$, 0.0 othw. $\ ),$
$\sup_a\{T(\ 1.0$ if $\mathsf{Place\_Name}(a, \mathsf{san\ francisco})$, 0.0 othw.,
$\max(\min(\frac{50\ \text{km}-d}{50\ \text{km}-20\ \text{km}}, 1), 0)\ )\}\ )\ )\}\ )\}$
| $\mathsf{Place\_Distance}(x, a, d) \wedge \mathsf{Place\_Population}(x, p)$

Figure 14: Semantic derivation for "`Carol lives in a small city near San Francisco`"

1. The syntactic token `Carol` was found in the input, as a result of the syntactic reduction `Nom` $\rightarrow$ `Carol`. The semantic production for this syntactic structure is given in the grammar as

$$\mathsf{Nom}(x) := 1.0 \text{ if } \mathsf{Person\_Name}(x, \mathsf{carol}),\ 0.0 \text{ otherwise.}$$

   The Nom in this rule resolves to node N_1, so we have

$$\mathsf{N}_1(x) := 1.0 \text{ if } \mathsf{Person\_Name}(x, \mathsf{carol}),\ 0.0 \text{ otherwise.}$$

   According to the example data in Figure 2 (Section 4.1), only $x = \mathsf{p}_1$ fulfills this, so the semantic representation of node N_1 is the fuzzy set

$$\mathsf{N}_1 := \{(\mathsf{p}_1, 1)\}.$$

2. The syntactic token `Carol` was found, as a result of the grammatical reduction `NP` $\rightarrow$ `Nom`. The semantic production for this syntactic structure is given in the grammar as

$$\mathsf{NP}(x) := \mathsf{Nom}(x).$$

   The NP in this rule resolves to node N_2, the Nom resolves to node N_1, so we have

$$\mathsf{N}_2(x) := \mathsf{N}_1(x),$$

   so the semantic representation of node N_2 is the fuzzy set

$$\mathsf{N}_2 := \{(\mathsf{p}_1, 1)\},$$

   the same as N_1.

3. The syntactic token `lives` was found in the input, as a result of the syntactic reduction `V` $\rightarrow$ `lives`. The semantic production for this syntactic structure is given in the grammar as

$$\mathsf{V}(x, \lambda_y, \lambda_z) := 1.0 \text{ if } \mathsf{Lives\_In}(x, \lambda_y, \lambda_z),\ 0.0 \text{ otherwise.}$$

   The V in this rule resolves to node N_3, so we have

$$\mathsf{N}_3(x, \lambda_y, \lambda_z) := 1.0 \text{ if } \mathsf{Lives\_In}(x, \lambda_y, \lambda_z),\ 0.0 \text{ otherwise.}$$

   This is fulfilled by several combinations $(x, \lambda_y, \lambda_z)$:

$$\begin{aligned}
x &= (\mathsf{p}_1, \mathsf{c}_{17}), \quad \lambda_y = \mathsf{p}_1, \quad \lambda_z = \mathsf{c}_{17}, \\
x &= (\mathsf{p}_2, \mathsf{c}_{54}), \quad \lambda_y = \mathsf{p}_2, \quad \lambda_z = \mathsf{c}_{54}, \\
x &= (\mathsf{p}_3, \mathsf{c}_{56}), \quad \lambda_y = \mathsf{p}_3, \quad \lambda_z = \mathsf{c}_{56}, \\
x &= (\mathsf{p}_4, \mathsf{c}_{57}), \quad \lambda_y = \mathsf{p}_4, \quad \lambda_z = \mathsf{c}_{57},
\end{aligned}$$

   so the semantic representation of node N_3 is the fuzzy relation

$$\begin{aligned}
\mathsf{N}_3 = \{\ & (((\mathsf{p}_1, \mathsf{c}_{17}), \mathsf{p}_1, \mathsf{c}_{17}), 1), \\
& (((\mathsf{p}_2, \mathsf{c}_{54}), \mathsf{p}_2, \mathsf{c}_{54}), 1), \\
& (((\mathsf{p}_3, \mathsf{c}_{56}), \mathsf{p}_3, \mathsf{c}_{56}), 1), \\
& (((\mathsf{p}_4, \mathsf{c}_{57}), \mathsf{p}_4, \mathsf{c}_{57}), 1)\ \}.
\end{aligned}$$

4. The syntactic token `a` was found in the input, as a result of the syntactic reduction `Det` $\rightarrow$ `a`. The semantic production for this syntactic structure is given in the grammar as

$$\mathsf{Det}(x) := 0.$$

The Det in this rule resolves to node $\mathsf{N_4}$, so we have

$$\mathsf{N_4}(x) := 0.$$

For the sake of simplicity we assumed, in this grammar, that determiners do not carry any meaning, therefore the semantic representation of node $\mathsf{N_4}$ is the empty fuzzy set, i.e. $\mathsf{N_4} = \emptyset$.

5. The syntactic token `small` was found in the input, as a result of the syntactic reduction `Adj` $\rightarrow$ `small`. The semantic production for this syntactic structure is given in the grammar as

$$\mathsf{Adj}(x) := \max(\min\left(\frac{20000 - p}{20000 - 10000}, 1\right), 0) \mid \mathsf{Place\_Population}(x, p).$$

The Adj in this rule resolves to node $\mathsf{N_5}$, so we have

$$\mathsf{N_5}(x) := \max(\min\left(\frac{20000 - p}{20000 - 10000}, 1\right), 0) \mid \mathsf{Place\_Population}(x, p).$$

This condition is fulfilled by the combinations

$$\begin{aligned} &x = \mathsf{c_1}, \quad p = \mathsf{43000}, \\ &x = \mathsf{c_2}, \quad p = \mathsf{101124}, \\ &\ldots \\ &x = \mathsf{c_{57}}, \quad p = \mathsf{183504}. \end{aligned}$$

A good example is $x = \mathsf{c_{17}}, p = \mathsf{12208}$ we can determine the degree of fulfillment directly by substituting into this function

$$\max(\min\left(\frac{20000 - 12208}{20000 - 10000}, 1\right), 0) = 0.7792$$

The other degrees of fulfillment can be determined analogously, so we get the semantic representation of node $\mathsf{N_5}$ as

$$\mathsf{N_5} = \{(\mathsf{c_1}, 0), (\mathsf{c_2}, 0), \ldots, (\mathsf{c_{17}}, 0.7792), \ldots, (\mathsf{c_{57}}, 0)\}.$$

6. The syntactic token `small` was found in the input, as a result of the syntactic reduction `AP` $\rightarrow$ `Adj`. The semantic production for this syntactic structure is given in the grammar as

$$\mathsf{AP}(x) := \mathsf{Adj}(x).$$

This resolves to

$$\mathsf{N_6}(x) := \mathsf{N_5}(x).$$

so we get

$$\mathsf{N_6} = \{(\mathsf{c_1}, 0), (\mathsf{c_2}, 0), \ldots, (\mathsf{c_{17}}, 0.7792), \ldots, (\mathsf{c_{57}}, 0)\}.$$

7. The syntactic token `city` was found in the input, as a result of the syntactic reduction `N` $\rightarrow$ `city`. The semantic production for this syntactic structure is given in the grammar as

$$\mathsf{N}(x) := 1.0 \text{ if } \mathsf{Place}(x), \text{ 0.0 otherwise.}$$

This resolves to

$$\mathsf{N}_7(x) := 1.0 \text{ if } \mathsf{Place}(x), \text{ 0.0 otherwise.}$$

This is fulfilled by

$$\begin{aligned} x &= \mathsf{c}_1, \\ x &= \mathsf{c}_2, \\ &\ldots, \\ x &= \mathsf{c}_{57}, \end{aligned}$$

so the semantic representation of node N_7 is the fuzzy set

$$\mathsf{N}_7 = \{(\mathsf{c}_1, 1), (\mathsf{c}_2, 1), \ldots, (\mathsf{c}_{57}, 1)\}.$$

8. The syntactic token `small city` was found in the input, as a result of the syntactic reduction `N'` $\rightarrow$ `AP N`. The semantic production for this syntactic structure is given in the grammar as

$$\mathsf{N'}(x) := T\Big(\mathsf{AP}(x), \mathsf{N}(x)\Big).$$

This resolves to

$$\mathsf{N}_8(x) := T\Big(\mathsf{N}_6(x), \mathsf{N}_7(x)\Big).$$

This is fulfilled by

$$\begin{aligned} x = \mathsf{c}_1 &: T\Big(\mathsf{N}_6(\mathsf{c}_1), \mathsf{N}_7(\mathsf{c}_1)\Big) = T\Big(0, 1\Big) = 0, \\ x = \mathsf{c}_2 &: T\Big(\mathsf{N}_6(\mathsf{c}_2), \mathsf{N}_7(\mathsf{c}_2)\Big) = T\Big(0, 1\Big) = 0, \\ &\ldots, \\ x = \mathsf{c}_{17} &: T\Big(\mathsf{N}_6(\mathsf{c}_{17}), \mathsf{N}_7(\mathsf{c}_2)\Big) = T\Big(0.7792, 1\Big) = 0.7792, \\ &\ldots, \\ x = \mathsf{c}_{57} &: T\Big(\mathsf{N}_6(\mathsf{c}_{57}), \mathsf{N}_7(\mathsf{c}_{57})\Big) = T\Big(0, 1\Big) = 0. \end{aligned}$$

Consequently, the semantic representation of node N_8 is the fuzzy set

$$\mathsf{N}_8 = \{(\mathsf{c}_1, 0), (\mathsf{c}_2, 0), \ldots, (\mathsf{c}_{17}, 0.7792), \ldots, (\mathsf{c}_{57}, 0)\}.$$

9. The syntactic token `San Francisco` was found in the input, as a result of the syntactic reduction `Nom` $\rightarrow$ `San Francisco`. The semantic production for this syntactic structure is given in the grammar as

$$\mathsf{Nom}(x) := 1.0 \text{ if } \mathsf{Place\_Name}(x, \mathsf{san\ francisco}), \text{ 0.0 otherwise.}$$

This resolves to

$$\mathsf{N}_9(x) := 1.0 \text{ if } \mathsf{Place\_Name}(x, \mathsf{san\ francisco}), \text{ 0.0 otherwise.}$$

This is fulfilled by

$$x = \mathsf{c}_{39}$$

so the semantic representation of node N_9 is the fuzzy relation

$$\mathsf{N}_9 = \{(\mathsf{c}_{39}, 1)\}.$$

10. The syntactic token `San Francisco` was found in the input, as a result of the syntactic reduction `NP` → `Nom`. The semantic production for this syntactic structure is given in the grammar as

$$\mathsf{NP}(x) := \mathsf{Nom}(x).$$

This resolves to

$$\mathsf{N}_{10}(x) := \mathsf{N}_9(x),$$

so we have

$$\mathsf{N}_{10} = \{(\mathsf{c}_{39}, 1)\}.$$

11. The syntactic token `near San Francisco` was found in the input, as a result of the syntactic reduction `PP` → `near NP`. The semantic production for this syntactic structure is given in the grammar as

$$\mathsf{PP}(x) := \sup_y \{T\Big(\mathsf{NP}(y), \max(\min\left(\frac{50\text{ km} - d}{50\text{ km} - 20\text{ km}}, 1\right), 0\Big) \mid \mathsf{Place\_Distance}(x, y, d)\}.$$

This resolves to

$$\mathsf{N}_{11}(x) := \sup_y \{T\Big(\mathsf{N}_{10}(y), \max(\min\left(\frac{50\text{ km} - d}{50\text{ km} - 20\text{ km}}, 1\right), 0\Big) \mid \mathsf{Place\_Distance}(x, y, d)\}.$$

This condition is fulfilled by the combinations

$$\begin{aligned}
&y = \mathsf{c}_1, \quad x = \mathsf{c}_2, \quad d = 537\text{ km},\\
&y = \mathsf{c}_1, \quad x = \mathsf{c}_3, \quad d = 460\text{ km},\\
&\ldots\\
&y = \mathsf{c}_{39}, \quad x = \mathsf{c}_1, \quad d = 554\text{ km},\\
&y = \mathsf{c}_{39}, \quad x = \mathsf{c}_2, \quad d = 59\text{ km},\\
&\ldots\\
&y = \mathsf{c}_{39}, \quad x = \mathsf{c}_{17}, \quad d = 33\text{ km},\\
&\ldots
\end{aligned}$$

A good example is $x = \mathsf{c}_{39}$, $y = \mathsf{c}_{17}$, $d = 33$ km. We can determine the degree of fulfillment directly by substituting into the function

$$T\Big(\mathsf{N}_{10}(y), \max(\min\left(\frac{50\text{ km} - d}{50\text{ km} - 20\text{ km}}, 1\right), 0)\Big) = T\left(1, \frac{50\text{ km} - 33\text{ km}}{50\text{ km} - 20\text{ km}}\right) = 0.566$$

Note that this will be the supremum over all y, since $\mathsf{N}_{10}(y) = 0$ for all y other than c_{39}. Doing this analogously for the other combinations we get

$$\begin{aligned}\mathsf{N}_{11} = \{\ &(\mathsf{c}_1, 0.0), (\mathsf{c}_2, 0.0), (\mathsf{c}_3, 0.0), \ldots, \\ &(\mathsf{c}_6, 1.0), \ldots, (\mathsf{c}_{17}, 0.566), \ldots, \\ &(\mathsf{c}_{28}, 1.0), \ldots, (\mathsf{c}_{39}, 0.0) \ldots, (\mathsf{c}_{57}, 0.0)\ \}.\end{aligned}$$

12. The syntactic token `small city near San Francisco` was found in the input, as a result of the syntactic reduction `N' → N' PP`. The semantic production for this syntactic structure is given in the grammar as

$$\mathsf{N'}(x) := T\Big(\mathsf{N'}(x), \mathsf{PP}(x)\Big).$$

This resolves to

$$\mathsf{N}_{12}(x) := T\Big(\mathsf{N}_8(x), \mathsf{N}_{11}(x)\Big).$$

This is fulfilled by

$$\begin{aligned}x = \mathsf{c}_1 &: T\Big(\mathsf{N}_8(\mathsf{c}_1), \mathsf{N}_{11}(\mathsf{c}_1)\Big) = T\Big(0, 0\Big) = 0, \\ x = \mathsf{c}_2 &: T\Big(\mathsf{N}_8(\mathsf{c}_2), \mathsf{N}_{11}(\mathsf{c}_2)\Big) = T\Big(0, 0\Big) = 0, \\ &\ldots, \\ x = \mathsf{c}_{17} &: T\Big(\mathsf{N}_8(\mathsf{c}_{17}), \mathsf{N}_{11}(\mathsf{c}_{17})\Big) = T\Big(0.7792, 0.566\Big) = 0.566, \\ &\ldots, \\ x = \mathsf{c}_{57} &: T\Big(\mathsf{N}_8(\mathsf{c}_{57}), \mathsf{N}_{11}(\mathsf{c}_{56})\Big) = T\Big(0, 0\Big) = 0.\end{aligned}$$

Consequently, the semantic representation of node N_{12} is the fuzzy set

$$\mathsf{N}_{12} = \{(\mathsf{c}_1, 0), (\mathsf{c}_2, 0), \ldots, (\mathsf{c}_{17}, 0.566), \ldots, (\mathsf{c}_{57}, 0)\}.$$

13. The syntactic token `a small city near San Francisco` was found in the input, as a result of the syntactic reduction `NP → Det NP`. The semantic production for this syntactic structure is given in the grammar as

$$\mathsf{NP}(x) := \mathsf{N'}(x).$$

This resolves to

$$\mathsf{N}_{13}(x) := \mathsf{N}_{12}(x).$$

Thus we have

$$\mathsf{N}_{13} = \{(\mathsf{c}_1, 0), (\mathsf{c}_2, 0), \ldots, (\mathsf{c}_{17}, 0.566), \ldots, (\mathsf{c}_{57}, 0)\}.$$

14. The syntactic token `in a small city near San Francisco` was found in the input, as a result of the syntactic reduction `PP → in NP`. The semantic production for this syntactic structure is given in the grammar as

$$\mathsf{PP}(x) := \mathsf{NP}(x).$$

This resolves to

$$\mathsf{N}_{14}(x) := \mathsf{N}_{13}(x).$$

Thus we have

$$\mathsf{N}_{14} = \{(\mathsf{c}_1, 0), (\mathsf{c}_2, 0), \ldots, (\mathsf{c}_{17}, 0.566), \ldots, (\mathsf{c}_{57}, 0)\}.$$

15. The syntactic token `lives in a small city near San Francisco` was found in the input, as a result of the syntactic reduction `VP` $\rightarrow$ `V PP`. The semantic production for this syntactic structure is given in the grammar as

$$\mathsf{VP}(x, \lambda_y) := \sup_z \{T\Big(\mathsf{V}(x, \lambda_y, z), \mathsf{PP}(z)\Big)\}$$

This resolves to

$$\mathsf{N}_{15}(x, \lambda_y) := \sup_z \{T\Big(\mathsf{N}_3(x, \lambda_y, z), \mathsf{N}_{14}(z)\Big)\}$$

This condition is fulfilled by the combinations

$$\begin{gathered}
z = \mathsf{c}_1, \quad x = (\mathsf{p}_1, \mathsf{c}_1), \quad \lambda_y = \mathsf{p}_1 : T\Big(\mathsf{N}_3((\mathsf{p}_1, \mathsf{c}_1), \mathsf{p}_1, \mathsf{c}_1), \mathsf{N}_{14}(\mathsf{c}_1)\Big) = 0, \\
z = \mathsf{c}_2, \quad x = (\mathsf{p}_1, \mathsf{c}_1), \quad \lambda_y = \mathsf{p}_1 : T\Big(\mathsf{N}_3((\mathsf{p}_1, \mathsf{c}_1), \mathsf{p}_1, \mathsf{c}_1), \mathsf{N}_{14}(\mathsf{c}_1)\Big) = 0, \\
\dots \\
z = \mathsf{c}_{17}, \quad x = (\mathsf{p}_1, \mathsf{c}_{17}), \quad \lambda_y = \mathsf{p}_1 : T\Big(\mathsf{N}_3((\mathsf{p}_1, \mathsf{c}_{17}), \mathsf{p}_1, \mathsf{c}_{17}), \mathsf{N}_{14}(\mathsf{c}_{17})\Big) = 0.566, \\
\dots
\end{gathered}$$

Note that, for $\lambda_y = \mathsf{p}_1$ we must get the supremum at $z = \mathsf{c}_{17}$, since $\mathsf{N}_3(x, \lambda_y, z)$ is zero for all other combinations. Proceeding analogously with all λ_y we get:

$$\begin{aligned}
\mathsf{N}_{15} = \{\ & (((\mathsf{p}_1, \mathsf{c}_1), \mathsf{p}_1), 0), \\
& (((\mathsf{p}_1, \mathsf{c}_2), \mathsf{p}_1), 0), \dots, \\
& (((\mathsf{p}_1, \mathsf{c}_{17}), \mathsf{p}_1), 0.566), \dots, \\
& (((\mathsf{p}_4, \mathsf{c}_{57}), \mathsf{p}_1), 0)\ \ \}.
\end{aligned}$$

16. The syntactic token `Carol lives in a small city near San Francisco` was found in the input, as a result of the syntactic reduction `S` $\rightarrow$ `NP VP`. The semantic production for this syntactic structure is given in the grammar as

$$\mathsf{S}(x) := \sup_y \{T\Big(\mathsf{NP}(y), \mathsf{VP}(x, y)\Big)\}$$

This resolves to

$$\mathsf{N}_{16}(x) := \sup_y \{T\Big(\mathsf{N}_2(y), \mathsf{N}_{15}(x, y)\Big)\}$$

This condition is fulfilled by the combinations

$$\begin{gathered}
y = \mathsf{p}_1, \quad x = (\mathsf{p}_1, \mathsf{c}_1) : T\Big(\mathsf{N}_2(\mathsf{p}_1), \mathsf{N}_{15}((\mathsf{p}_1, \mathsf{c}_1), \mathsf{p}_1)\Big) = 0, \\
y = \mathsf{p}_1, \quad x = (\mathsf{p}_1, \mathsf{c}_2) : T\Big(\mathsf{N}_2(\mathsf{p}_1), \mathsf{N}_{15}((\mathsf{p}_1, \mathsf{c}_2), \mathsf{p}_1)\Big) = 0, \\
\dots \\
y = \mathsf{p}_1, \quad x = (\mathsf{p}_1, \mathsf{c}_{17}) : T\Big(\mathsf{N}_2(\mathsf{p}_1), \mathsf{N}_{15}((\mathsf{p}_1, \mathsf{c}_{17}), \mathsf{p}_1)\Big) = 0.566, \\
\dots
\end{gathered}$$

Note that, for $y = \mathsf{p}_1$ we must get the supremum at $x = (\mathsf{p}_1, \mathsf{c}_{17})$, since $\mathsf{N}_{16}(x, y)$ is zero for all other x. Proceeding analogously with all y we get:

$$\begin{aligned}
\mathsf{N}_{16} = \{\ & ((\mathsf{p}_1, \mathsf{c}_1), 0), \\
& ((\mathsf{p}_1, \mathsf{c}_2), 0), \dots, \\
& ((\mathsf{p}_1, \mathsf{c}_{17}), 0.566), \dots, \\
& ((\mathsf{p}_4, \mathsf{c}_{57}), 0)\ \ \}.
\end{aligned}$$

So we determined for each element in our domain a degree to which it matches the meaning of the English sentence `Carol lives in a small city near San Francisco`, which is exactly what we wanted to find.

B The prototype

B.1 placesdb.pl

```
place( k(c,1) ).
place_name( k(c,1) ) --> [altadena].
place_lat( k(c,1), lat(n, ang(34,11,22,99)) ).
place_long( k(c,1), long(w, ang(118,7,49,1)) ).
place_pop( k(c,1), pop(43000) ).

place( k(c,2) ).
place_name( k(c,2) ) --> [antioch].
place_lat( k(c,2), lat(n, ang(38,0,18,0)) ).
place_long( k(c,2), long(w, ang(121,48,17,1)) ).
place_pop( k(c,2), pop(101124) ).

...

place( k(c,51) ).
place_name( k(c,51) ) --> [vallejo].
place_lat( k(c,51), lat(n, ang(38,6,15,0)) ).
place_long( k(c,51), long(w, ang(122,15,20,2)) ).
place_pop( k(c,51), pop(119708) ).

place( k(c,52) ).
place_name( k(c,52) ) --> [walnut,creek].
place_lat( k(c,52), lat(n, ang(37,54,22,99)) ).
place_long( k(c,52), long(w, ang(122,3,50,0)) ).
place_pop( k(c,52), pop(65151) ).

place( k(c,53) ).
place_name( k(c,53) ) --> [new,york].
place_lat( k(c,53), lat(n, ang(40,42,51,1)) ).
place_long( k(c,53), long(w, ang(74,0,23,2)) ).
place_pop( k(c,53), pop(8085742) ).

place( k(c,54) ).
place_name( k(c,54) ) --> [los,angeles].
place_lat( k(c,54), lat(n, ang(34,3,8,0)) ).
place_long( k(c,54), long(w, ang(118,14,34,2)) ).
place_pop( k(c,54), pop(3819951) ).

place( k(c,55) ).
place_name( k(c,55) ) --> [tokyo].
place_lat( k(c,55), lat(n, ang(35,41,10,0)) ).
```

```
place_long( k(c,55), long(e, ang(139,45,9,0)) ).
place_pop( k(c,55), pop(12064100) ).

place( k(c,56) ).
place_name( k(c,56) ) --> [cambridge].
place_lat( k(c,56), lat(n, ang(52,12,56,16)) ).
place_long( k(c,56), long(e, ang(0,5,56,8)) ).
place_pop( k(c,56), pop(131465) ).

place( k(c,57) ).
place_name( k(c,57) ) --> [linz].
place_lat( k(c,57), lat(n, ang(48,15,28,0)) ).
place_long( k(c,57), long(e, ang(14,15,46,95)) ).
place_pop( k(c,57), pop(183504) ).
```

B.2 placeskb.pl

```
ang2rad( ang( Deg, Min, Sec, HSec ), Rad ) :-
  Rad is (Deg + Min/60 + Sec/(60*60) + HSec/(60*60*100)) * pi/180.

lat2rad(n, Ang, Rad ) :-
  ang2rad( Ang, AngDec ),
  Rad is pi/2 - AngDec.

lat2rad(s, Ang, Rad ) :-
  ang2rad( Ang, AngRad ),
  Rad is pi/2 + AngRad.

long2rad(e, Ang, Rad ) :-
  ang2rad( Ang, Rad ).

long2rad(w, Ang, Rad ) :-
  ang2rad( Ang, AngRad ),
  Rad is -AngRad.

spheric_distance(LatA, LongA, LatB, LongB, X) :-
  X is 6371 * acos( sin(LatA) * sin(LatB) * cos( LongA-LongB ) +
                    cos(LatA) * cos(LatB) ).

place_distance( A, B, dist( Dist ) ) :-
  place( A ),
    place_lat( A, lat( LatNSA, LatA ) ),
    lat2rad( LatNSA, LatA, LatRadA ),
    place_long( A, long( LongEWA, LongA ) ),
    long2rad( LongEWA, LongA, LongRadA ),
  place( B ),
    place_lat( B, lat( LatNSB, LatB ) ),
    lat2rad( LatNSB, LatB, LatRadB ),
    place_long( B, long( LongEWB, LongB ) ),
```

```
    long2rad( LongEWB, LongB, LongRadB ),
  spheric_distance(
    LatRadA, LongRadA,
    LatRadB, LongRadB,
    Dist ).
```

B.3 peopledb.pl

```
person( k(p,1) ).
person_name( k(p,1) ) --> [carol].

person( k(p,2) ).
person_name( k(p,2) ) --> [frank].

person( k(p,3) ).
person_name( k(p,3) ) --> [richard].

person( k(p,4) ).
person_name( k(p,4) ) --> [john].

lives_in( k(r, k(p,1), k(c,17)) ).
lives_in( k(r, k(p,2), k(c,54)) ).
lives_in( k(r, k(p,3), k(c,56)) ).
lives_in( k(r, k(p,4), k(c,57)) ).

likes( k(l, k(p,1), k(p,3)) ).
likes( k(l, k(p,1), k(p,4)) ).
```

B.4 db.pl

```
domain(X) :- place(X).
domain(X) :- person(X).
domain(X) :- lives_in(X).
domain(X) :- likes(X).
```

B.5 fuzzylogic.pl

```
tnorm(D,X,Y):-D is float(min(X,Y)).
tnorm(D,[X1,X2]):-tnorm(D,X1,X2).
tnorm(D,[X1,X2|Xs]):-tnorm(Y,[X2|Xs]),tnorm(D,X1,Y).

supr(D,X,Y):-D is float(max(X,Y)).
supr(D,[X1,X2|Xs]):-supr(Y,[X2|Xs]),supr(D,X1,Y).
supr(D,[D]).

atlFuzSet( D, X, Xmin,Xcen ) :-
  D is max( min( (X-Xmin)/(Xcen-Xmin), 1 ), 0).

atmFuzSet( D, X, Xcen,Xmax ) :-
```

```
  D is max( min( (Xmax-X)/(Xmax-Xcen), 1 ), 0).

trapFuzSet( D,X, Xmin,XcenA,XcenB,Xmax ) :-
  atlFuzSet( D, X, Xmin, XcenA ), atmFuzSet( D, X, XcenB, Xmax ).

triFuzSet( D, X, Xmin,Xcen,Xmax ) :-
  trapFuzSet( D,X, Xmin,Xcen,Xcen,Xmax ).
```

B.6 language.pl

```
sent( SentD, X, A, [] ) :-
  domain( X ),
  findall( DoF, Y^sentx( DoF, X, Y, A, [] ), DoFs ),
  supr( SentD, DoFs ).
sentx( SentD, X, Y ) -->
  np( NpD, Y ),
  vp( VpD, X, Y ),
  { tnorm( SentD, NpD, VpD ) }.


vp( VpD, X, Y, A, [] ) :-
  domain( X ), domain( Y ),
  findall( DoF, Z^vpx( DoF, X, Y, Z, A, [] ), DoFs ),
  supr( VpD, DoFs ).
vpx( VpD, X, Y, Z ) -->
  v( VD, X, Y, Z ), pp( PpD, Z ),
  { tnorm( VpD, VD, PpD ) }.
vpx( VpD, X, Y, Z ) -->
  v( VD, X, Y, Z ), np( NpD, Z ),
  { tnorm( VpD, VD, NpD ) }.


v( 1.0, k(r, Y, Z), Y, Z ) --> [lives],
  { lives_in( k(r, Y, Z) ), person( Y ), place( Z ) }.

v( 1.0, k(l, Y, Z), Y, Z ) --> [likes],
  { likes( k(l, Y, Z) ), person( Y ), person( Z ) }.


np( NpD, X ) --> nom( NpD, X ).
np( NpD, X ) --> det, nb( NpD, X ).


nb( D, X ) --> nb1( D, X ).
nb( D, X ) --> nb2( D, X ).

nb2( NbD, X ) --> n( NbD, X ).

nb2( NbD, X ) --> ap( ApD, X ), n( ND, X ),
```

```
  { tnorm( NbD, [ApD, ND] ) }.

nb1( NpD, X ) --> nb2( NppD, X ), pp( PpD, X ),
  { tnorm( NpD, [NppD, PpD] ) }.


pp( PpD, X ) --> [in], np( PpD, X ),
  { place( X ) }.


pp( PpD, X, A, [] ) :-
  domain( X ),
  findall( DoF, Y^ppx( DoF, X, Y, A, [] ), DoFs ),
  supr( PpD, DoFs ).
ppx( PpD, X, Y ) --> [near], np( NpD, Y ),
  { place( X ), place( Y ), X \= Y,
    place_distance( X, Y, dist( Dist ) ),
    atmFuzSet( PD, Dist, 20, 50 ),
    tnorm( PpD, PD, NpD ) }.


ap( ApD, X ) --> [very], ap( AppD, X ),
  { ApD is AppD*AppD }.

ap( ApD, X ) --> [fairly], ap( AppD, X ),
  { ApD is sqrt(AppD) }.

ap( ApD, X) --> adj( ApD, X ).


det --> [the].
det --> [a].


adj( AD, X ) --> [huge],
  { place( X ),
    place_pop( X, pop( Pop ) ),
    atlFuzSet( AD, Pop, 1000000, 3000000 ) }.

adj( AD, X ) --> [large],
  { place( X ),
    place_pop( X, pop( Pop ) ),
    atlFuzSet( AD, Pop, 300000, 500000 ) }.

adj( AD, X ) --> [small],
  { place( X ),
    place_pop( X, pop( Pop ) ),
    atmFuzSet( AD, Pop, 10000, 20000 ) }.
```

```
adj( AD, X ) --> [tiny],
  { place( X ),
    place_pop( X, pop( Pop ) ),
    atmFuzSet( AD, Pop, 100, 1500 ) }.


n( 1.0, X ) --> [city],
  { place(X) }.

n( 1.0, X ) --> [town],
  { place(X) }.


nom( 1.0, X ) --> place_name( X ),
  { place( X ) }.

nom( 1.0, X ) --> person_name( X ),
  { person( X ) }.
```

B.7 main.pl

```
#!/usr/bin/pl -s

:-consult(fuzzylogic).

:-consult(placesdbsrt).
:-consult(placeskb).

:-consult(peopledbsrt).

:-consult(db).

:-consult(language).
```

B.8 A Dialog with the Prototype

```
%  fuzzylogic compiled 0.00 sec, 2,568 bytes
%  placesdbsrt compiled 0.01 sec, 35,208 bytes
%  placeskb compiled 0.00 sec, 2,516 bytes
%  peopledbsrt compiled 0.00 sec, 2,408 bytes
%  db compiled 0.00 sec, 636 bytes
%  language compiled 0.01 sec, 7,508 bytes
% ./main.pl compiled 0.02 sec, 53,216 bytes
Welcome to SWI-Prolog (Multi-threaded, Version 5.1.13)
Copyright (c) 1990-2003 University of Amsterdam.
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software,
and you are welcome to redistribute it under certain conditions.
Please visit http://www.swi-prolog.org for details.
```

```
For help, use ?- help(Topic). or ?- apropos(Word).

?- nom(D, X, [san,francisco], []).

D = 1.0
X = k(c, 39) ;

No
?- nom(D, X, [carol], []).

D = 1.0
X = k(p, 1) ;
?- np(D, X, [a,city], []).

D = 1.0
X = k(c, 1) ;

...
D = 1.0
X = k(c, 57) ;

No
?- np(D, X, [a,small,city], []).

D = 0.0
X = k(c, 1) ;

...
D = 0.0
X = k(c, 10) ;

D = 0.9967
X = k(c, 11) ;

...
D = 0.0
X = k(c, 57) ;

No
?- np(D, X, [a,very,small,city], []).

D = 0.0
X = k(c, 1) ;

...
D = 0.0
X = k(c, 10) ;

D = 0.993411
X = k(c, 11) ;
```

```
...

D = 0.0
X = k(c, 57) ;

No
?- np(D, X, [a,very,very,small,city], []).

D = 0.0
X = k(c, 1) ;

...

D = 0.0
X = k(c, 10) ;

D = 0.986865
X = k(c, 11) ;

...

D = 0.0
X = k(c, 57) ;

No
?- pp( D, X, [near,a,small,city], [] ).

D = 0.0
X = k(c, 1) ;

...

D = 0.922049
X = k(c, 10) ;

D = 1.0
X = k(c, 11) ;

...

D = 0.0
X = k(c, 57) ;

No
?- sent( D, X, [carol,lives,in,a,small,city,near,san,francisco], [] ).

D = 0.7792
X = k(r, k(p, 1), k(c, 17)) ;

No
?- sent( D, X, [carol,lives,in,a,very,small,city,near,a,\
                big,city,near,san,francisco], [] ).
```

```
D = 0.607153
X = k(r, k(p, 1), k(c, 17)) ;

No
?- sent( D, X, [carol,lives,near,a,very,small,city,near,a,\
                big,city,near,san,francisco], [] ).

D = 0.779021
X = k(r, k(p, 1), k(c, 17)) ;

No
```