# Generating Argumentative Judgment Determiners

**Michael Elhadad**
Ben Gurion University of the Negev
Dept of Mathematics and Computer Science
Beer Sheva, 84105, Israel
elhadad@bengus.bgu.ac.il *

## Abstract

This paper presents a procedure to generate *judgment determiners, e.g., many, few.* Although such determiners carry very little objective information, they are extensively used in everyday language. The paper presents a precise characterization of a class of such determiners using three semantic tests. A conceptual representation for sets is then derived from this characterization which can serve as an input to a generator capable of producing judgment determiners. In a second part, a set of syntactic features controlling the realization of complex determiner sequences is presented. The mapping from the conceptual input to this set of syntactic features is then presented. The presented procedure relies on a description of the speaker's *argumentative intent* to control this mapping and to select appropriate judgment determiners.

## Introduction

There are cases when answering *many* is a sign of ignorance:

Teacher: How many neutrons are there in an atom of Uranium?
Child: many...

In other cases, though, uttering a precise number is of no help to the hearer:

Q: how difficult is Topology 101?
A1: It has six assignments.
A2: It requires many assignments.

In A1, the precise number of assignments in the class can be seen as an awful lot or a pretty average workload. In all cases, the precise number does not satisfy the communicative need expressed by the question, and answer A2, with a determiner like *many* is more felicitous. This paper addresses the issue of producing such *judgment determiners* (JDs) in a text generation system, focusing on a class I call *argumentative judgment determiners.*

This problem has been mostly ignored in previous work in generation for two reasons: first, most of the previous work on determiner generation has focused on the difficult decision definite/indefinite; second, most existing generation systems, except for Dale's EPICURE (Dale [1988]), do not focus on the issue of non-singular NPs. Consequently, the generation of JDs, although it fulfills an important pragmatic function, has remained largely unexplored.

The determiner generation procedure presented here is implemented as part of ADVISOR II, a generation system which provides advice to university students preparing their course schedule (Elhadad [1993]). In this domain, an analysis of a corpus of 40,000 words containing transcripts of recordings of advising sessions with human academic advisors shows the following distribution of determiners:

| Determiner type | # Occurrences |
|---|---|
| Article (a, the) | 1540 |
| Demonstrative (this, that) | 950 |
| Cardinal | 210 |
| Judgment det | 300 |

In this table, judgment dets include *(many, few, all, no, a lot, a large number of, lots of).* This distribution indicates that, at least in this domain, whenever a quantity must be referred to, JDs are used more often than exact determiners, and highlights the need to cover JDs in a generation system like ADVISOR II.

The paper starts by defining JDs and provides a semantic characterization of JDs. I derive from this characterization a set of requirements on the form of the input representation that must be sent to a generator to allow it to produce JDs. I then discuss the syntax of judgment determiners and explain how the generator maps the input conceptual representation of sets to a set of syntactic features controlling the selection of JDs.

*Reprinted from *Proceedings of the 11th National Conference on Artificial Intelligence (AAAI 93)*, MIT Press, 1993, pp.344-349. This paper reports on work pursued while the author was at Columbia University, Dept of Computer Science

## Semantic Characterization of Judgment Determiners

Observing that *242* and *many* do not satisfy the same pragmatic function, one gets the intuition that *many* is a member of a "different" class of determiners - ones that do not express only objective information. This section uses three semantic properties defined in (Barwise & Cooper [1981]) and (Keenan & Stavi [1986]) in order to precisely identify the class of *judgment determiners* (JDs) and derive constraints on the form of the input required by a generator to produce JDs.

### Non Extensionality and Argumentation

The first property of JDs is that they are *non-extensional*, in the sense defined in (Keenan & Stavi [1986], p.257):

> To say that a det $d$ is extensional is to say, for example, that whenever the doctors and the lawyers are the same individuals then *d doctors* and *d lawyers* have the same properties, *e.g.*, *d doctors attended the meeting* necessarily has the same truth value as *d lawyers attended the meeting.*

The following example shows that *many* for example, is not extensional:

> Imagine that in the past the annual doctors meeting has been attended by tens of thousands of doctors, and only two or three lawyers. But, during the course of the year, and unbeknownst to everyone, all the doctors get law degrees and all the lawyers get medical degrees (so that doctors and lawyers are now the same) and at this year meeting only 500 doctors/lawyers show up. Reasonably then (a) is true and (b) is false:
> (a) Many lawyers attended the meeting this year.
> (b) Many doctors attended the meeting this year.
> Thus *many (few)* cannot be treated extensionally. (Keenan & Stavi [1986], pp.257-8)

Keenan & Stavi, therefore, propose to consider an expression such as *many Xs* as "simply indeterminate in truth value." In other words, using a non-extensional determiner such as *many Xs* does not say much about the number of $X$s, but instead, expresses *a decision by the speaker* to highlight the number of $X$s as significant. The input to a generator must therefore record this decision if any non-extensional determiner is to be produced.

The work presented here uses the notion of *argumentative intent* to account for this speaker's decision. An argumentative intent is the goal to convince a hearer of a certain conclusion. Following Anscombre & Ducrot ([1983]), it is hypothesized that simple evaluations, of the form *(X is high/low on scale S)*, and simple argumentative rules, of the form *(the higher X is on P, the higher Y is on Q)* are sufficient to account for many linguistic phenomena related to argumentation. In previous work, I have discussed the impact of the speaker's argumentative intent on different generation tasks: content selection and organization (Elhadad [1992]), connective selection (Elhadad & McKeown [1990]), adjective selection (Elhadad [1991]). The same general mechanism can be applied to the selection of JDs. In this case, I assume that the generator's input includes argumentative evaluations of the form *(X is high/low on S)* where $X$ is a finite set of discrete individuals and $S$ is the scale of cardinality. When this is the case, a feature *degree* is set in the description of the set $X$, which records the speaker's argumentative intent regarding the number of elements in the set $X$.

### Monotonicity: the orientation feature

Barwise & Cooper define the notion of *monotonicity* using the following linguistic test (Barwise & Cooper [1981], pp.184-191): consider two verb-phrases $VP_1$ and $VP_2$, such that the denotation of $VP_1$ is a subset of the denotation of $VP_2$, that is, in logical terms, $VP_1(x) \Rightarrow VP_2(x)$. Then by checking whether the following seem logically valid, one can determine if the determiners are monotonic:

> If NP $VP_1$, then NP $VP_2$.
> (NP is monotonic increasing.)
> If NP $VP_2$, then NP $VP_1$.
> (NP is monotonic decreasing.)

Barwise & Cooper give the following examples, taking $VP_1$ to be *entered the race early* and $VP_2$ to be *entered the race* (Barwise & Cooper [1981], p.185):

$$\text{If} \left\{ \begin{array}{ll} \textit{some} & \textit{Republican} \\ \textit{every} & \textit{linguist} \\ \textit{John} & \\ \textit{most} & \textit{farmers} \\ \textit{many} & \textit{men} \end{array} \right\} \text{entered the race early,}$$

$$\left\{ \begin{array}{ll} \textit{some} & \textit{Republican} \\ \textit{every} & \textit{linguist} \\ \textit{John} & \\ \textit{most} & \textit{farmers} \\ \textit{many} & \textit{men} \end{array} \right\} \text{entered the race.}$$

All these implications are valid, while the reverse implications do not hold. Similarly, the following implications indicate that the determiners *no, few* and *neither* are monotonic decreasing:

$$\text{If} \left\{ \begin{array}{ll} \textit{no} & \textit{plumber} \\ \textit{few} & \textit{linguists} \\ \textit{neither} & \textit{Democrat} \end{array} \right\} \text{entered the race,}$$

$$\left\{ \begin{array}{ll} \textit{no} & \textit{plumber} \\ \textit{few} & \textit{linguists} \\ \textit{neither} & \textit{Democrat} \end{array} \right\} \text{entered the race early.}$$

Note that the determiners *exactly two* and *at most three* are not monotonic at all, since there is no implicative relation between *exactly three men entered the race early* and *exactly three men entered the race.*

All argumentative JDs must be monotonic. The feature *orientation* is required in the input specification of a set to indicate the orientation of an argumentative evaluation. Its value can be +, - or none, and it corresponds exactly to the distinction between monotonic increasing, decreasing, and non-monotonic quantifiers. Note that orientation is distinct from degree, because different degrees can be expressed for the same orientation:

| | |
|---|---|
| *AI has a little programming*: | `orient + degree -` |
| *AI has a lot of programming*: | `orient + degree +` |
| *AI has little programming*: | `orient - degree -` |
| *AI has almost no programming*: | `orient - degree +` |

### The Intersection Condition

Following (Barwise & Cooper [1981], Sect.1.3), NPs *as a whole* are viewed as the expression of *generalized quantifiers*, as opposed to simply determiners like *all* or *some*. Consequently, the input to the determiner generation procedure is a complete set specification. Sets are characterized in intension by a domain and the properties that must be satisfied by all elements. These properties are in general mapped to modifiers of the NP realizing the set using a procedure similar to that discussed in (Elhadad [1993]). This section presents the *intersection condition*, defined in (Barwise & Cooper [1981], p.190) and explains why a distinction between two types of modifiers must be enforced to allow for the generation of JDs.

The linguistic test corresponding to the formal definition of the *intersection condition* is the following: let $P_1$ and $P_2$ be two properties; then if a determiner $D$ satisfies the intersection condition, the sentences *there are* $D$ $P_1$ $P_2$ $N$ and $D$ $P_1$ $N$ *are* $P_2$ are semantically equivalent. For example:

There are exactly 3 interesting AI topics.
Exactly 3 interesting topics are in AI.
Exactly 3 AI topics are interesting.

These three forms are equivalent, indicating that *exactly n* satisfies the intersection condition. In contrast, consider:

(1) There are many interesting topics which are in AI.
(2) There are many AI topics which are interesting.

These NPs are not equivalent, as shown, for example, by considering the following situation: a person has interest in 100 topics; AI covers 10 topics; the intersection between the interesting topics and the AI topics contains 7 elements. Then (1) is probably not valid (7 topics out of 100 is not many) while (2) is valid (7 out of 10 is many). Note that the "classical" quantifiers, corresponding to the mathematical $\exists$ and $\forall$, both satisfy the intersection condition, but JDs, *e.g., many, few, most*, do not satisfy it.

Consider now the fact that in both (1) and (2) the NPs with the *many* determiner denote the same set of individuals (the 7 topics of the intersection). The validity of the sentences, however, is different when the scope of the *many* changes from one modifier to the other. This indicates again that *many* is not extensional, but also, that the input conceptual description of sets must attribute a different status to the two modifiers if modifier generation is to interact properly with determiner selection and prevent the generation of invalid sentences like (1).

I distinguish between *reference* and *intension* modifiers to account for this difference in status. For example, consider the set defined by:
$S = \{x \in \text{TOPICS} \mid \textit{Interest(x,student)} \wedge \textit{Area(x,AI)}\}$
Different perspectives can be held on this set: when the *interest* property is the intension and the *area* property is the reference, the definition can be written as follows:
$S1 = \{x \in \text{AI-TOPICS} \mid \textit{Interest(x,student)}\}$
And, under normal circumstances this representation leads to the English realization: *Most AI topics are interesting.*

If in contrast the perspective is switched, and *interest* becomes the reference and *area* the intension, then the definition and realization become:
$S2 = \{x \in \text{INTERESTING-TOPICS} \mid \textit{Area(x,AI)}\}$
*Few of the topics that interest you are in AI.*
In this example, the same observation of a set of topics satisfying two properties can lead to the generation of two contradictory argumentative evaluations. This indicates that, because JDs do not satisfy the intersection condition, the structuring of properties in a set specification between *reference* and *intension* must be present in the input to the generator (as in S1 and S2), and that a neutral representation for sets such as S would not be appropriate.

In summary, the following three properties characterize argumentative JDs: (1) they are non-extensional; (2) they are monotonic; (3) they do not satisfy the intersection condition. Consequently, the conceptual description of sets sent as input to a generator must contain the features *degree* and *orientation* and distinguish between *reference* and *intension* modifiers if the generator is to be able to produce JDs.

## Input/Output

The overall architecture of the generation part of ADVISOR II is the following: the input is a conceptual representation encoded in a KL-ONE-like network enriched with pragmatic annotations describing the speaker's intentions and assumptions. This conceptual network is passed to a *lexical chooser* which selects open-class words and performs *phrase planning* to combine them into phrase structures such as NPs and clauses. These structures are finally passed to the *syntactic realization grammar* SURGE for closed-class word selection, agreements and linearization. In this paper, I only describe the *determiner selection* subprocess of the lexical chooser.

The input to the determiner generation procedure, therefore, is a set specification. The output is a set of syntactic features appearing at the NP level and controlling the selection of the determiner sequence in the SURGE grammar.

### Conceptual Representation for Generalized Quantifiers

ADVISOR II is implemented in FUF, an extension of the functional unification formalism of Kay ([1979]) described in (Elhadad [1993], Chap.3 and 4). This section describes the conceptual representation of sets as a FUF functional description input to the generator. The input specification contains objects of four types: individuals, sets, relations and argumentative evaluations. I briefly present here the representation of sets and evaluations.

**Sets** are described by the following features (all are optional except `cat` and `index`):

```
((cat set)
 (index <unique-id>)
 (kind <prototype>)
 (cardinality <n>)
 (extension <list-of-individuals>)
 (intension <a-relation>)
 (reference <a-set>))
```

`kind` is used for sets of objects of the same type. `extension` is the explicit list of the set elements. The logical definition of a set described by `intension` and `reference` is the following:
$S = \{x \in \text{ Reference } \mid \text{Intension}(x)\}$ where `intension` is a relation, and `reference`, recursively, a set and the distinction between `intension` and `reference` is justified above.

**Argumentative evaluations** encode the speaker's argumentative intent:

```
((cat evaluation)
 (evaluated <path-to-set-or-individual>)
 (scale <a-scale>)
 (orientation <+ or ->))
```

This indicates that the speaker judges the element pointed to by `evaluated` as high (or low) on `scale`.

An input for the following set is shown below with the argumentative evaluation that the set is high on the scale of cardinality:
$S1 = \{x \in \text{AI-TOPICS} \mid Interest(x, student)\}$ Intuitively, this set contains the 7 topics that are of interest to the user among the 10 topics which are covered in AI.

```
((topics
  ((cat set) (kind ((cat topic)))
   (cardinality 7)
   (reference
    ((cat set) (kind ((cat topic)))
     (cardinality 10)
     (intension
      ((cat class-relation) (name area)
       (1 {^ argument})
       (2 ((cat field) (name AI)))))))
   (intension
    ((cat user-relation) (name interest)
     (1 {^ argument})
     (2 ((cat student)))))))
 (argumentation
  ((cat evaluation) (evaluated {topics})
   (scale ((name cardinality)))
          (orientation +))))
```

## Output: Syntax of the Determiner Sequence

The determiner sequence is a subconstituent of NPs. It is also in itself a complex constituent. It has the specificity that it is mainly a *closed system - i.e.*, the lexical elements are part of a small set of words which are determined completely by a small set of syntactic features. When implementing the SURGE realization grammar, the issue was to identify a minimal set of features accounting for the variety of determiner sequences observable in English. The syntactic description implemented in SURGE is an augmented version of that presented in (Halliday [1985], pp.159-176), with additions derived from observations in (Quirk et al. [1972], pp.136-165). A set of 24 features controlling the realization of the determiner sequence was thus identified, which is presented in detail in (Elhadad [1993], Sect.5.4). I only present here a brief overview of the grammar for determiners, and focus on the features relevant to the realization of JDs.

| pre-det | (of) | det | deictic2 | ord | card |
|---|---|---|---|---|---|
| *all* | *of* | *the* | *famous* | *first* | *ten* |
| *half* | *of* | *my* | | | |
| *twice as* | | | | | |

| quant | NP-head |
|---|---|
| | *commandments* |
| *many* | *properties* |
| *much* | *work* |

Figure 1: Syntactic structure of the NP

The structure of the determiner sequence is shown in Fig.1. Pre-determiners can be one of the following elements: *all, both* or *half*, multipliers (*twice, three times*) or fractions (*one fourth*). Complex co-occurrence restrictions exist between the different predeterminers and different classes of nouns (mass, count nouns denoting a number or an amount) and between predeterminers, cardinals and quantifiers. There are also special cases of noun classes that take zero articles, including seasons, institutions, transport means, illnesses (Quirk et al [1972], pp.156-159). The implementation of such co-occurrence restrictions explains the complexity of SURGE's determiner grammar.

To control the selection of the various elements of the determiner sequence, I make use of Halliday's distinction between three functions of the determiner sequence:

**1. Deictic**: to identify whether a subset of the thing is denoted, and if yes, which subset. The relevant decisions are depicted below, in Fig.2, in the form of a systemic network, where curly braces indicate choice points between alternatives and square brackets indicate simultaneous decisions which must be taken. The top level distinction is between specific and non-specific determination. A specific deictic denotes a known, well identified subset of the thing. A non-specific deictic denotes a subset identified by quantity.

For specific deictics, the subset can be identified by different means: deixis and distance (near or far from

- *specific*
  - *demonstrative* — *this*
  - *possessive* — *my*
  - *the* — *the*
  ---
  - *determinative* — *this*
  - *interrogative* — *which*
  ---
  - *nonplural*
    - *singular*
    - *mass*
  - *plural*
- *nonspecific*
  - *total*
    - *positive* — *all*
    - *negative* — *no*
  - *partial*
    - *selective*
    - *nonselective*
    ---
    - *degree*
    - *evaluative*
    - *unmarked*
    ---
    - *singular*
    - *nonsingular*
    - *unmarked*

Figure 2: The deictic network

the speaker - *this* vs. *that*), possession (*my, John's*) or not at all (*the*).

Non-specific deictics are either total (*all, no, both, each, every, neither*) or partial. Partial deictics come in two sorts: selective (*one, either, any* and *some* as in *some people*) and non-selective (*a, some* as in *some cheese*).

**2. Deictic2**: to specify the subset of the thing by "referring to its fame, familiarity or its status in the text" (Halliday [1985], p.162). The deictic2 element is an adjective such as *same, usual, given*. Such adjectives are part of the determiner sequence because they systematically occur before the cardinal element of the determiner, in contrast to any other describing adjective, which must occur after the cardinal.

**3. Numerative**: to specify the quantity or the place of the thing. The numerative specification can be either quantitative (expressing a quantity, *three*) or ordinative (expressing a relative position, *third*). In both cases, the expression can be either exact (*one, two..., first, second...*) or inexact (*a lot, the next*). The source of the inexactness can be an approximation device (*about three, roughly third, approximately ten*) or a range expression (*between six and ten*). Alternatively, it can be a context dependent expression like *the next, many, few, more*, and an evaluative expression like *enough, too many, too much*. Figure 3 summarizes the relevant decisions.

The features controlling the selection of JDs are located in the non-specific region of the deictic network and in the inexact region of the numerative network. The subset of SURGE features which trigger the selection of argumentative JDs is `total, orientation, superlative` and `degree`.

- *quantitative* — three, a lot
- *ordinative* — third, next
---
- *exact* — three, third
- *inexact*
  - *indefinite* — a few, many
  - *fuzzy* — about three
  - *range* — between 6 and 8
  - *comparative* — more, less
  - *multiplier* — twice
  - *fraction* — half
  - *context* — many, enough

Figure 3: The numerative network

## Mapping from Conceptual to Syntactic Features

When mapping from a conceptual description to the features controlling the determiner selection, the first decision is whether the speaker's argumentative intent is to be realized through the use of a JD or with other linguistic devices (such as connotative verbs, scalar adjectives or connectives). This decision can interact with most generation decisions and is discussed at length in (Elhadad [1993]).

When argumentation *is* to be expressed in a determiner site, the following mapping rules are applied:

- **Total**: when the set is the object of a positive evaluation, its cardinality is known and equal to that of the reference set, then total is set to +. If the evaluation is negative and the cardinality is known to be 0, total is set to -. In all other cases, total is set to none.
- **Orientation**: when the set is the object of an argumentative evaluation, orientation records whether the evaluation is high or low. Otherwise, it is set to none.
- **Superlative**: set to yes when the reference set is given, its cardinality is known, the cardinality of the set is larger than half that of the reference set, and the set is the object of a positive argumentative evaluation.

The general heuristics behind these rules is to use the pragmatically strongest determiner possible to realize the speaker's argumentative intent. For example, if *all* *AI topics are interesting* can be produced, it will be preferred to *some* *AI topics*.

For **Degree**, the determination of a value is more difficult. Degree determines the selection among *a few, some, many, a (large, great, incredible...) number* if orientation is +, and among *few, a (small, tiny, ridiculous...) number* if orientation is -. In ADVISOR II, degree is limited to have values +, - or none. A finer account of the degree of determiners is probably needed, but it creates many problems which, for lack of space, cannot be discussed here.

## Conclusion

This paper has presented a method to generate judgment determiners (JDs). It focuses on the use of JDs as one way (among many others) to express the speaker's argumentative intent. The paper provides a semantic characterization of JDs through the use of three tests (non-extensionality, monotonicity and non-satisfaction of the intersection condition) and derives constraints from this characterization on the form of input a generator requires to be capable of producing JDs.

The paper describes the part of a lexical chooser that takes as input a conceptual description of a set with pragmatic annotations such as argumentative evaluations and produces as output a set of syntactic features which control the behavior of the SURGE surface realization component. The component of SURGE responsible for the complex syntax of the English determiner sequence is discussed and a technique to map the conceptual input to the relevant set of features is presented.

**Acknowledgments.** I am indebted to Kathleen McKeown, Jacques Robin and Rebecca Passonneau for their precious help both during the research and the writing of this paper.

## References

J. C. Anscombre and O. Ducrot. *L'argumentation dans la langue.* Pierre Mardaga, Bruxelles, 1983.

J. Barwise and R. Cooper. Generalized quantifiers in english. *Linguistics and Philosophy*, 4:159–219, 1981.

R. Dale. *Generating referring expressions in a domain of objects and processes.* PhD thesis, University of Edinburgh, Scotland, 1988.

M. Elhadad. Types in functional unification grammars. In *Proceedings of the 28th Annual Meeting of the Association for Computational Linguistics*, Detroit, MI, 1990. ACL.

M. Elhadad. Generating adjectives to express the speaker's argumentative intent. In *Proceedings of the 9th Annual Conference on Artificial Intelligence.* AAAI, 1991.

M. Elhadad. Generating argumentative paragraphs. In *Proceedings of COLING'92*, Nantes, France, July 1992.

M. Elhadad. *Using argumentation to control lexical choice: a unification-based implementation.* PhD thesis, Computer Science Department, Columbia University, 1992.

M. Elhadad. Generating complex noun phrases. Technical Report FC-93-05, Dept of Mathematics and Computer Science, Ben Gurion University of the Negev, Israel.

M. Elhadad and K. R. McKeown. Generating connectives. In *Proceedings of COLING'90 (Volume 3)*, pages 97–101, Helsinki, Finland, 1990.

M. Halliday. *An introduction to functional grammar.* Edward Arnold, London, 1985.

M. Kay. Functional grammar. In *Proceedings of the 5th Annual Meeting of the Berkeley Linguistic Society*, 1979.

E. Keenan and Y. Stavi. A semantic characterization of natural language determiners. *Linguistics and Philosophy*, 9:253–326, 1986.

R. Quirk, S. Greenbaum, G. Leech, and J. Svartvik. *A grammar of contemporary English.* Longman, 1972.