CONTEXT SENSITIVE GENERATION OF DESCRIPTIONS

Emiel Krahmer and Mariët Theune

IPO, Center for Research on User-System Interaction, Eindhoven, The Netherlands
{krahmer/theune}@ipo.tue.nl

ABSTRACT

Probably the best current algorithm for generating definite descriptions is the Incremental Algorithm due to Dale and Reiter. If we want to use this algorithm in a Concept-to-Speech system, however, we encounter two limitations: (*i*) the algorithm is insensitive to the linguistic context and thus always produces the same description for an object, (*ii*) the output is a list of properties which uniquely determine one object from a set of objects: how this list is to be expressed in spoken natural language is not addressed. We propose a modification of the Incremental Algorithm based on the idea that a definite description refers to the most salient element in the current context satisfying the descriptive content. We show that the modified algorithm allows for the context-sensitive generation of both distinguishing and anaphoric descriptions, while retaining the attractive properties of Dale and Reiter's original algorithm.

1. INTRODUCTION

In their interesting 1995 paper, Dale & Reiter present various algorithms they developed alone or in tandem to determine the content of a *distinguishing description,* that is: a definite description which is an accurate characterization of the entity being referred to, but not of any other object in the current 'context set' (taken to be the set of objects speaker and hearer are currently attending to). They argue that their *Incremental Algorithm* is the best one from a computational point of view (it is fast) as well as from a psychological point of view (humans appear to do it in a similar way).

However, if we want to use this algorithm in a *Concept-to-Speech* system, we encounter two problems. First, the algorithm is not sensitive to the linguistic context. Consider:

(1) a. Al bought the small grey poodle and the blue siamese cat.
    b. Unlike the cat, the dog was a bargain.

In (1)a, two distinguishing descriptions are used: *the small grey poodle* and *the blue siamese cat.* If a speaker wants to refer to these animals again, she typically will not repeat the distinguishing descriptions, but rather use anaphoric, reduced descriptions which, as it were, are distinguishing *given the linguistic context*. It would not be very attractive to define a separate algorithm for such anaphoric descriptions, which would operate alongside the Incremental Algorithm. Fortunately, this is not necessary; it turns out that with some modifications we can use the Incremental Algorithm for both kinds of definite descriptions. The second aspect of the Incremental Algorithm which prohibits immediate embedding in a Concept-to-Speech system is that its output is not a natural language expression, but a list of properties which uniquely selects one object from a context set. The question how this list of properties is to be expressed in spoken natural language is not addressed. Still, when developing a Concept-to-Speech system that is what we are interested in: we want to know which properties realized in the definite description should be accented due to the fact that they are *new* to the discourse, or since they stand in a *contrast* relation (as is the case with the properties *dog* and *cat* in example (1)b). This second problem is solved as a side-effect of embedding the proposed modified algorithm in *LGM*, a language and domain independent Concept-to-Speech system discussed in Klabbers et al. (1998, *these proceedings*, see also van Deemter & Odijk 1997, Theune et al. 1997). The modifications to the Incremental Algorithm which we want to argue for are based on the idea that definite descriptions refer to the *most salient* element satisfying the descriptive content (Lewis 1979: 348-350, formalized in Krahmer 1998). Lewis mentions the following example (due to McCawley):

(2) The dog got in a fight with another dog.

Lewis notes that this statement can only be true in a domain which contains at least two dogs, which entails that *the dog* cannot be a distinguishing description. According to Lewis, (2) means that the most salient dog got in a fight with some less salient dog. Lewis does not mention descriptions which refer to 'unique' objects (i.e., distinguishing descriptions), but it is readily seen that they can also be understood in terms of salience: if there is only one object with the relevant properties, it *has* to be the most salient one.

2. THE INCREMENTAL ALGORITHM

Let us first discuss the Incremental Algorithm. For the sake of illustration we use domain D_1 from Dale & Reiter (1995:258):

d_1	⟨ type, chihuahua ⟩, ⟨ size, small ⟩, ⟨ colour, black ⟩
d_2	⟨ type, chihuahua ⟩, ⟨ size, large ⟩, ⟨ colour, white ⟩
d_3	⟨ type, siamese cat ⟩, ⟨ size, small ⟩, ⟨ colour, black ⟩

Following Dale & Reiter (1995:254) we make the following basic assumptions about domains: (*i*) each entity in the domain is characterized by a list of attribute value pairs, or *properties,* (*ii*) each entity has at least an attribute *type* and (*iii*) there may be a subsumption hierarchy on certain attributes, generally on 'type'. For instance: 'dog' subsumes 'chihuahua', 'cat' subsumes 'siamese cat' and the type 'animal' in its turn subsumes 'dog' and 'cat'. Additionally, we assume that the *basic level values* (Rosch 1978) for 'chihuahua' and 'siamese cat' are 'dog' and 'cat' respectively.

The input for the Incremental Algorithm is the object r for which a distinguishing description is to be constructed, together with a set C of alternative objects from which r has to be distinguished ('dis-

Let U_k be a sentence uttered in the context of sw_i, let $\textsf{topic}(U_k) \subseteq D$ and $\textsf{focus}(U_k) \subseteq D$ be the sets of entities which are referred to in the topic and the focus of U_k respectively. Then the new salience function sw_{i+1} is defined as follows:

$$sw_{i+1}(d) = \begin{cases} 10 & \text{if } d \in \textsf{focus}(U_k) \\ 9 & \text{if } d \in \textsf{topic}(U_k) \text{ and } d \text{ is referred to by a definite NP} \\ sw_i(d) & \text{if } d \in \textsf{topic}(U_k) \text{ and } d \text{ is referred to by a pronoun} \\ \textsf{max}(sw_i(d) - 1, 0) & \text{if } d \notin \textsf{topic}(U_k) \cup \textsf{focus}(U_k) \text{ and } d \in \textsf{topic}(U_l), l < k \\ \textsf{max}(sw_i(d) - 2, 0) & \text{if } d \notin \textsf{topic}(U_k) \cup \textsf{focus}(U_k) \text{ and } d \in \textsf{focus}(U_l), l < k \end{cases}$$

Figure 1: Salience Weight assignment based on Hajičová (1993). Roughly, the topic of sentence U, $\textsf{topic}(U)$, is what sentence U is about, while the focus, $\textsf{focus}(U)$, is what the sentence says about its topic. Notice that all information which is required to compute the topic-focus dichotomy (which information is recoverable from the discourse context and which information is not) is directly available in LGM.

tractors') and, crucially, an ordered set of preferred attributes. This list contains, in order of preference, the attributes that human speakers and hearers prefer for a particular domain. For instance, we may assume that a human speaker would first try to describe an animal by its 'type' (is it a dog? is it a cat?), and if that doesn't help attributes like 'colour' and 'size' may be used. It is reasonable to assume that speakers have a general preference for *absolute* properties such as 'colour', which are easily observed without taking the other objects into account, over *relative* properties such as 'size', which are less easily observed and always require inspection of the distractors. Thus let us assume that the list of preferred attributes for the example domain is $\langle$ type, colour, size, ...$\rangle$. Essentially, the Incremental Algorithm goes through this list, and for each attribute it encounters, it determines the best value of this attribute (i.e., the one closest to the basic-level value) and checks whether it rules out one or more of the remaining distractors. If so, this best value is added to the set of properties which will be used in the generation of the distinguishing description. The algorithm stops when the list of distractors is empty (success) or when the end of the list of preferred attributes is reached (failure).

Discussion A noteworthy feature is that there is no backtracking (hence the term 'incremental'): once a property p has been selected, it will be realised in the final description, even if a property which is added later would render the inclusion of p redundant 'with hindsight'. This aspect is partly responsible for the efficiency of their algorithm, but Dale & Reiter additionally claim that this property is 'psychologically realistic' since human speakers also often include redundant modifiers in their referring expressions. They write: *"For example, in a typical experiment a participant is shown a picture containing a white bird, a black cup, and a white cup and is asked to identify the white bird; in such cases, participants generally produce the referring expression* the white bird, *even though the simpler form* the bird *would have been sufficient.* (Dale & Reiter 1995:248). We would like to make two remarks. First, it seems to us that the Incremental Algorithm produces the description *the bird* in this situation: if we make the natural assumption that 'type' is the most preferred attribute, the property 'bird' will be the first one selected and immediately rules out the black and the white cup. The second point concerns the experimental paradigm described here, which goes back to Pechmann (1984). In these experiments, the described situation is typically preceded by a situation in which the participant is shown a picture containing an additional black bird which the subject described as *the black bird*. It has been argued that consequently the adjective *white* in *the white bird* is not 'unnecessary', but that it serves to signal a contrast of the current object with another object in the discourse model/linguistic context. In other words: this indicates that the creation of the referring expression is *context-sensitive*.

3. A MODIFIED ALGORITHM

The underlying idea of the modifications we want to make to the Incremental Algorithm is the following: *A definite description* 'the $\overline{\text{N}}$' *is a suitable description of an object d in a state s iff d is the most salient object with the property expressed by $\overline{\text{N}}$ in state s.* We first introduce some notational conventions. Let L be the list of properties expressed by some $\overline{\text{N}}$. The *value set* of L in some domain D (notation $\textsf{Val}_D(L)$) is the set of objects $d \in D$ which have the properties expressed by L. Thus, $\textsf{Val}_{D_1}(\{\langle \text{size}, \text{small}\rangle\}) = \{d_1, d_3\}$. When omitting the domain subscript and the attribute does not lead to confusion, we just write $\textsf{Val}(\text{small})$. By definition, the value set of the empty list of properties is the entire domain ($\textsf{Val}_D(\{\}) = D$). Following common practice, we use $|S|$ to denote the cardinality of a set S.

How can we model the salience of an entity? For that purpose we use a function variable sw ('salience weight') which represents a total function mapping each element in the domain to a natural number from 0 (*not salient*) to 10 (*maximally salient*). A central question is how an object increases in salience. Krahmer & Theune (1998) formalize and compare two methods to assign salience weights, one based on the hierarchical focusing constraints of Hajičová (1993), the other on the Centering approach of Grosz et al. (1995). Our intention is not so much to argue that one of these approaches is superior to the other. Rather, we want to show that the modified algorithm can be associated with at least two realistic ways of determining salience weights. For the sake of illustration, the one based on Hajičová (1993) is given in figure 1. For a more detailed discussion we refer to Krahmer and Theune (1998). Now we can state that an object r is the most salient object having certain properties L in a state s (notation: $\textsf{MostSalient}(r, L)$) if, and only if, every object in $\textsf{Val}(L)$ different from r has a lower salience weight than r itself.

Figure 2 contains our proposal for a modified algorithm in pseudocode. To ease comparison, we have stuck as closely as possible to Dale & Reiter (1995:257) in both notation and structure. Below, we illustrate the algorithm in figure 2 with a number of examples. First, we give a general, somewhat informal overview. The algorithm is called by $\textsf{MakeReferringExpression}\ (r, P)$; that is, we try to generate a definite description for a referent r given some pre-defined list P of *preferred attributes*. L is a list of the properties which have been selected for inclusion in the expression generated and is initialized as the empty list. The variable $tree$ contains the syntactic tree for the $\overline{\text{N}}$ under construction which corresponds with the current list of properties L. Finally, $contrast$ is a boolean variable which indicates whether the property under consideration is contrastive or not. As in the original version the main

```
$\boxed{\text{MakeReferringExpression } (r, P)}$
$L \leftarrow \{\}$
$tree \leftarrow$ nil
$contrast \leftarrow$ **false**
**for** each member $A_i$ of list $P$ **do**
    $V \leftarrow$ FindBestValue $(r, A_i, \text{BasicLevelValue}(r, A_i))$
    **if** $|\text{Val}(L \cup \{\langle A_i, V \rangle\})| < |\text{Val}(L)| \wedge$
      $contrast \leftarrow \text{Contrastive}(r, A_i, V) \wedge$
      $(tree \leftarrow \text{UpdateTree}(tree, V, contrast)) \neq$ nil
    **then** $L \leftarrow L \cup \{\langle A_i, V \rangle\}$
    **endif**
    **if** MostSalient $(r, L) =$ **true**
    **then if** $\langle$type$, X\rangle \in L$ for some $X$
            **then return** $[_{\text{NP}}[_{\text{Det}}\text{the}]\ tree]$
            **else** $V \leftarrow \text{BasicLevelValue}(r, \text{type}) \wedge$
                $(tree \leftarrow \text{UpdateTree}(tree, V, \textbf{false})) \neq$ nil $\wedge$
                **return** $[_{\text{NP}}[_{\text{Det}}\text{the}]\ tree]$
            **endif**
  **endif**
**return failure**

$\boxed{\text{FindBestValue}(r, A, \textit{initial-value})}$
**if** UserKnows$(r, \langle A, \textit{initial-value}\rangle) =$ **true**
**then** *value* $\leftarrow$ *initial-value*
**else** *value* $\leftarrow$ nil
**endif**
**if** MostSalient$(r, \{\langle A, \textit{value}\rangle\}) =$ **false** $\wedge$
  $(\textit{more-specific-value} \leftarrow \text{MoreSpecificValue}(r, A, \textit{value})) \neq$ nil $\wedge$
  $(\textit{new-value} \leftarrow \text{FindBestValue}(r, A, \textit{more-specific-value})) \neq$ nil $\wedge$
  $|\text{Val}(\{\langle A, \textit{new-value}\rangle\})| < |\text{Val}(\{\langle A, \textit{value}\rangle\})|$
**then** *value* $\leftarrow$ *new-value*
**endif**
**return** *value*

$\boxed{\text{MostSalient } (r, L)}$
**if** $\forall d \in \text{Val}(L) : d \neq r \Rightarrow sw(d) < sw(r)$
**then return true**
**else return false**
**endif**

$\boxed{\text{Contrastive}(r, A, V)}$
$C \leftarrow \{d \in \text{DR}(\text{PrevS} \cup \text{CurrS}) \mid d \neq r\ \wedge$
        Parent (BasicLV (d, type)) = Parent (BasicLV (r, type))$\}$
**if** $\exists d \in C : \text{Value}(d, A) \neq V$
**then return true**
**else return false**
**endif**
```

Figure 2: Full sketch of the modified algorithm. In the clause Contrastive (r, A, V), BasicLevelValue is abbreviated as BasicLV. DR(PrevS $\cup$ CurrS) is the set of objects referred to in the current or the previous sentence. A full description of UpdateTree is not given. It differs from the core algorithm in that it is, to a large extent, domain and language dependent. It attempts to integrate each new property V in the syntactic tree for the $\overline{\text{N}}$ constructed so far. In general, the 'type' attribute is realised as the head noun, and further pre- and postmodifiers are added in order of selection, provided the result is grammatically correct. If $contrast$ is **true**, the expression of the property V is marked by a [+c] feature, which is taken into account by the AddProsody function of LGM see Klabbers et al. (1998, *these proceedings*).

loop iterates through the list P. For each property (attribute value pair) on this list, the best value V is sought (essentially in the manner described above). Once the best value V is found, it is checked whether adding this property to the list of already selected properties 'shrinks' the value set (and thus rules out one or more 'distractors'). If this is so, it is checked whether V is contrastive. A property V of r is considered to be contrastive if there is an object referred to in the current or the previous sentence which is of the same kind as r but has a different value for the current attribute.[1] Subsequently, the algorithm tries to incorporate the property V in the $\overline{\text{N}}$ under construction, using the function UpdateTree $(tree, V, contrast)$. If this does not succeed (the lexical or syntactic restrictions of the generation module make it impossible to express the property), V is rejected. If it *does* succeed, the current property is added to the list of selected properties. Then it is checked whether the intended referent r is the most salient object in the current state of the discourse which satisfies L. If so, the algorithm succeeds, and the constructed $\overline{\text{N}}$ is combined with the definite determiner *the* to produce a full definite description.

First example: non-anaphoric description Reconsider our example domain D_1. Suppose we want to generate an expression for d_2 in a situation where all objects are equally salient. We call MakeReferringExpression (d_2, P) (with $P = \langle$ type, colour, size, ...$\rangle$).[2] We consider the first property of d_2, $\langle$ type, chihuahua $\rangle$. The best value for this attribute is 'dog' (since $\text{Val}(\text{chihuahua}) = \text{Val}(\text{dog}) = \{d_1, d_2\}$).[3] It is easily seen that This property has sufficient descriptive content to be included in the description under construction: $|\text{Val}(\text{dog})| = 2 < |\text{Val}(\{\})| = 3$. As a result the function UpdateTree is called which returns a simple tree consisting of an $\overline{\text{N}}$ with an N^0 *dog*. The value of L is now $\{\langle$ type, dog $\rangle\}$. MostSalient(d_2, dog) fails (because d_1 is also a dog, and both d_1 and d_2 are equally salient by assumption.). So we proceed by taking the second property of d_2, $\langle$ colour, white $\rangle$. Now we have $|\text{Val}(\text{white}, \text{dog})| < |\text{Val}(\text{dog})|$; this property is discriminating and again we call the function UpdateTree which updates the current $\overline{\text{N}}$ tree (which only contained the head noun *dog*) and adds an AP to this tree for the property 'white'. Now, MostSalient$(d_2, \{\text{white}, \text{dog}\})$ *is* true: d_2 is the only white dog in the domain D_1 so it is by definition the most salient one: the resulting tree for *the white dog* is returned. Notice that when we assume that all entities in the domain are always equally salient (thus sw represents the constant function mapping each $d \in D$ to some n), the modified algorithm selects exactly the same properties as the original Incremental Algorithm. In other words, our version indeed generalizes the original, which brings us to the next example.

Second example: anaphoric description (1) Suppose we continue where the first example ended: we refer to d_2 in a situation where d_2 is more salient than the other objects in the domain. Again, we call MakeReferringExpression (d_2, P). The first property encountered is $\langle$ type, chihuahua $\rangle$, and the best value

[1] Thus, loosely speaking, the property 'large' in the NP *the large dog* is contrastive in the context of *a small cat* but not in the context of *a small car*. This treatment of contrast is related to the proposal of Prevost (1996), who presents an algorithm for deciding which properties should receive contrastive accent in a manner which is somewhat similar to the Incremental Algorithm. See Theune (1997) for some further discussion.

[2] We assume, for the sake of simplicity, that the linguistic context is empty, which entails that $contrast$ will remain **false** in this and the following two examples.

[3] Notice that 'dog' is the BasicLevelValue for d_2 with respect to the attribute 'type'.

for this attribute is 'dog'. Again $|\mathsf{Val}(\text{dog})| < |\mathsf{Val}(\{\})|$. But now $\mathsf{MostSalient}(d_2, \text{dog})$ is *true*: $\mathsf{Val}(\text{dog}) = \{d_1, d_2\}$ and the only element in this value set different from d_2 has a lower salience weight than d_2. So, the result is (a tree for) *the dog*. Thus: if d_2 is more salient than all other dogs in the domain, we can subsequently use the description *the dog* to refer back to it. Of course, Dale & Reiter's version, being insensitive to differing contexts, would always express d_2 using the properties 'dog' and 'white'.

Third example: anaphoric description (2) An anaphoric description generally contains less information than its antecedent. In the previous example this was reflected by the omission of the property 'white' when d_2 was referred to a second time. But it may also happen that a more general head noun is used for subsequent reference. To show how our algorithm captures this intuition, let us add a poodle to the domain, switching to D_2, which is $D_1 + \{d_4, \langle$ type, poodle $\rangle$, $\langle$ size, small$\rangle$, $\langle$ colour, white $\rangle$ $\}$. If we call MakeReferringExpression (d_2, P) in a situation where all elements of D_2 have an equal salience weight, the result will now not be *the white dog*, as in the first example, but *the white chihuahua* (as the reader may want to check). If a *subsequent* call of MakeReferringExpression (d_2, P) occurs, d_2 will be more salient than the other elements of D_2 . The BasicLevelValue for d_2 with attribute 'type' is 'dog', and since d_2 is the currently most salient dog in the domain, this basic value is the best value as well. Thus, initial reference to d_2 (in domain D_2) yields *the white chihuahua* and subsequent reference, *the dog* (on the assumption that the initial reference is not accompanied by references to other dogs). Dale & Reiter's algorithm would always express d_2 (for domain D_2) using the properties 'white' and 'chihuahua'.

Fourth example: contrast Now consider calling the function MakeReferringExpression (d_2, P) (again for domain D_2), when the preceding sentence referred to both d_1 and d_2 (e.g., '*the black chihuahua and the white chihuahua* …'). Then the first property added to the description under construction will be 'chihuahua'. Now we consider the second property, 'white'. This property will also be added to the description, since it distinguishes d_2 from d_1. Moreover, this property is also *contrastive*, since d_1 and d_2 have the same basic level value for the 'type' attribute and different values for the 'colour' attribute. The result is '*the white*$^{[+c]}$ *chihuahua*', where the [+c] feature indicates a contrast relation. The LGM function AddProsody (Klabbers et al. 1998, *these proceedings*) takes this feature into account when it assigns pitch accents to a sentence. How a 'contrastive intonation' is best realised is discussed in Krahmer & Swerts (1998, *these proceedings*).

4. CONCLUDING REMARKS

We have discussed a generalization of Dale & Reiter's Incremental Algorithm which differs from the original version in that the linguistic context is taken into account, success is defined in terms of salience and the output is a syntactic NP tree with markers for contrastiveness. Klabbers et al. (1998, *these proceedings*) show how the modified algorithm can be embedded in LGM and how as a side-effect of this embedding, the referring expressions are 'dressed up' with prosodic information. As we have seen, our version of the Incremental Algorithm is a real generalization of the Dale & Reiter version, of which the attractive features have been retained. First, our algorithm can be shown to have the same theoretical complexity as Dale & Reiter's (polynomial), and if anything, requires less computational effort since the use of context enables a reduction of the number of attributes mentioned in the final referring expression. Second, even though we are reluctant to claim psychological reality, the modified algorithm might be argued to be at least as 'psychologically plausible' as Dale & Reiter's original, since it does more justice to examples like (1), illustrating that the creation of a referring expression is co-determined by the linguistic context.

Acknowledgements Krahmer was partly supported by the EC Language Engineering Program, project LE-1 2277 (VODIS). Theune did her work within the framework of the Priority Programma Language and Speech Technology (TST), sponsored by NWO (The Netherlands Organization for Scientific Research). Thanks are due to Jan Landsbergen, Jacques Terken and Marc Swerts.

5. REFERENCES

1. Dale, R., and Reiter, E. "Computational Interpretations of the Gricean Maxims in the Generation of Referring Expressions," *Cognitive Science* 18: 233-263, 1995.
2. van Deemter, K., and Odijk, J., "Context Modelling and the Generation of Spoken Discourse," *Speech Communication* 21(1/2):101-121, 1997.
3. Grosz, B., Joshi, A., and Weinstein, S., "Centering: A Framework for Modeling the Local Coherence of Discourse," *Computational Linguistics* **21**(2):203-225, 1995.
4. Hajičová, E., "Issues of Sentence Structure and Discourse Patterns," *Theoretical and Computational Linguistics*, Vol 2, Prague, Charles University, 1993.
5. Klabbers, E., Krahmer, E., and Theune, M., "A Generic Algorithm for Generating Spoken Monologues," 1998, *these proceedings*.
6. Krahmer, E., *Presupposition and Anaphora*, CSLI Publications, Stanford, 1998.
7. Krahmer, E., and Theune, M., *Generating Descriptions in Context*, MS, IPO, Eindhoven, 1998, (http://www.tue.nl/ipo/people/theune/).
8. Krahmer, E., and Swerts, M., "Reconciling Two Competing Views on Contrastiveness," 1998, *these proceedings*.
9. Lewis, D., "Scorekeeping in a Language Game," *Journal of Philosophical Logic*, 8:339-359, 1979.
10. Pechmann, T., *Überspezifizierung und Betonung in Referentieller Kommunikation*, doctoral dissertation, Mannheim University, 1984.
11. Prevost, S., "An Information Structural Approach to Spoken Language Generation," *Proceedings of the 34th ACL*, Santa Cruz, 294-301, 1996.
12. Rosch, E. (1978), Principles of Categorization, *Cognition and Categorization*, E. Rosch and B. Lloyd (eds.), Erlbaum, 27-48.
13. Theune, M., "Contrastive Accent in a Data-to-Speech System," *Proceedings of the 35th ACL/EACL*, Madrid, 519-521, 1997.
14. Theune, M., Klabbers, E., Odijk, J. and de Pijper, J.R., *From Data to Speech: A Generic Approach*, MS, IPO, 1997, (http://www.tue.nl/ipo/people/theune/manuscript.ps.Z).