Distinguishable Entities: Definition and Properties

Monique Rolbert
Laboratoire d'Informatique
Fondamentale de Marseille,
LIF, CNRS UMR 6166,
Aix-Marseille Université,
Marseille, France
monique.rolbert@lif.univ-mrs.fr

Pascal Préa
Laboratoire d'Informatique
Fondamentale de Marseille,
LIF, CNRS UMR 6166,
École Centrale Marseille,
Marseille, France
pprea@ec-marseille.fr

Abstract

Many studies in natural language processing are concerned with how to generate definite descriptions that evoke a discourse entity already introduced in the context. A solution to this problem has been initially proposed by Dale (1989) in terms of distinguishing descriptions and distinguishable entities. In this paper, we give a formal definition of the terms "distinguishable entity" in non trivial cases and we show that its properties lead us to the definition of a distance between entities. Then, we give a polynomial algorithm to compute distinguishing descriptions.

1 Introduction

Many studies in natural language processing are concerned with how to generate definite descriptions that evoke a discourse entity already introduced in the context (Dale, 1989; Dale and Haddock, 1991; Dale and Reiter, 1995; van Deemter, 2002; Krahmer et al., 2002; Gardent, 2002; Horacek, 2003), and more recently (Viethen and Dale, 2006; Gatt and van Deemter, 2006; Croitoru and van Deemter, 2007). Following Dale (1989), these definite descriptions are named "distinguishing descriptions". Informally, a distinguishing description is a definite description which designates one and only one entity among others in a context set. Conversely, this entity is named "distinguishable entity".

Things are simple if all the properties of the entities are unary relations. Let's give a set of entities $E = \{e_1, e_2\}$ with the following properties:
e_1: red, bird ; e_2: red, bird, eat ;

e_1 is not a distinguishable entity because there exists no distinguishing description that could designate e_1 and not e_2[1]. e_2 is a distinguishable entity and could be designated by the distinguishing description "the red bird that is eating".

Many of the works cited above are concerned with how to generate the best distinguishing description with the best algorithm, essentially in the unary case, that is if entities properties are unary ones. They focus on the length or the relevance of the generated expressions, or on the efficiency of the algorithm. But none of them give a formal definition of these "distinguishable entities". They all use an intuitive definition, more or less issued from the unary case and that could be resumed as follow: *an entity e is a distinguishable entity in E if and only if there exists a set of properties of e that are true of e and of no other entity in E.*

Unfortunately, this intuitive definition does not apply as it is in non-unary cases. The main problem comes with the notion of "set of properties of e": what is the set of properties of an entity if non-unary relations occur? Let us see this problem on an example. Suppose that we have an entity b_1 that is a bowl and that is on an entity t_1 which is a table. The set of entities is $E = \{b_1, t_1\}$ with:
b_1: bowl ; t_1: table ; on(b_1, t_1)

What is the set of properties of b_1? Dale and Haddock (1991) and, more or less, Gardent (2002), suggest that the property set for an entity includes all the relations in which it is involved (even non unary ones), and no others. Following this definition, the set of properties of b_1 should be {bowl(b_1), on(b_1, t_1)}.

Now, what if there is another bowl (b_2), which is on a table (t_2)? The set of properties of b_2 is {bowl(b_2), on(b_2, t_2)}, which is different from that

[1] One could object that "the red bird that is not eating" is a distinguishing description for e_1. But we do not make the Closed World Assumption ("every thing that is not said is false"). So, negative properties have to appear explicitly, like positive one, in entities description; their treatment causes no particular problem in our model

Proceedings of the 12th European Workshop on Natural Language Generation, pages 34–41,
Athens, Greece, 30 – 31 March 2009. ©2009 Association for Computational Linguistics

of b_1. But does it follow that b_1 is distinguishable from b_2? If the "intuitive definition" is used, the answer is *yes*: the set of properties of b_1 (and the formula $(\lambda x\, bowl(x) \wedge on(x, t_1)))$ is true for b_1 and for no other entity in $E = \{b_1, b_2, t_1, t_2\}$. But, one can immediately see that the "right" answer should depend on what we know about t_1 and t_2. If the only thing we know is that t_1 and t_2 are tables, then there is no definite description that designates b_1 and not b_2, and thus b_1 is not distinguishable from b_2. So, even if the formula on(-, t_1) is formally different from the formula on(-, t_2) and b_1 satisfies the first one and not the second one, that does not imply that b_1 is distinguishable from b_2.

So, the fact is that to determine if b_1 is distinguishable from b_2, knowing that the set of properties of b_1 is true for b_1 and not for b_2 is not sufficient: we have to determine if t_1 is distinguishable from t_2. That clearly leads to a non-trivial recursive definition and non-trivial recursive processes.

Two recent works describe algorithms that deal with this problem (Krahmer et al., 2003; Croitoru and van Deemter, 2007). Their works are both based on graph theory and their algorithms deal well with the non-unary case, but their computations need exponential time.

In this paper, our main goal is to give a definition of a distinguishable entity which corresponds to the intuitive sense and which works well even in non-trivial cases. Then we study its properties, which leads us to an interesting notion of distance between entities. Finally, we give a polynomial algorithm able to produce a distinguishing description whenever it is possible and which is based on this definition.

2 A definition of "distinguishable entity"

Intuitively, an entity e_1 is distinguishable from an entity e_2 in two cases:

- e_1 involves properties that are not involved by e_2 (we will say that e_1 is *0-distinguishable* from e_2)
- otherwise, e_1 and e_2 are in relations (we will precisely see how below) with at least two distinguishable entities e'_1 and e'_2. In this case, we will say that e_1 is *$(k+1)$-distinguishable* from e_2 if e'_1 is *k-distinguishable* from e'_2.

Basically, a *property* is an n-ary relation, together with a rank (the argument's position). For instance, with the fact e_1 *eats* e_2, e_1 has the property eat with rank 1 (noted eat_1) and e_2 has the property eat_2. So, e_1 and e_2 do not have the same set of properties. Conversely, if e_1 eats X and e_2 eats Y, e_1 and e_2 involve the same property (eat_1).

For an entity e, we denote $\mathcal{P}(e)$ the set of its properties. We will say that a tuple $t = (x_1, \ldots, x_p)$ *matches* a property p_q with e if $p(x_1, \ldots, x_{q-1}, e, x_q, \ldots, x_p)$ is true.

Definition 1 *(k-distinguishability D_k):*
*An entity e_1 is **0-distinguishable** from an entity e_2 (we denote it $e_1\, D_0\, e_2$) if $\mathcal{P}(e_1)$ is not included in $\mathcal{P}(e_2)$.*
*An entity e_1 is **k-distinguishable** $(k > 0)$ from an entity e_2 (we denote it $e_1\, D_k\, e_2$) if there exists a relation R_q in $\mathcal{P}(e_1)$ and a tuple $(x_1, \ldots, x_p)$ such that:*

- *$(x_1, \ldots, x_p)$ matches R_q with e_1.*
- *For every $(y_1, \ldots, y_p)$ that matches R_q with e_2, there exists some x_i and some $k' < k$ such that x_i is k'-distinguishable from y_i.*

We remark that if $e_1\ D_k\ e_2$, then $e_1\ D_j\ e_2$, for every $j > k$. So, we can define the more general notion of distinguishability (without a rank).

Definition 2 *(distinguishability D):*
*We say that an entity e_1 is **distinguishable from** an entity e_2 (we denote it $e_1\, D\, e_2$) if it is k-distinguishable from e_2, for some $k \geq 0$.*
*We say that e is **distinguishable** in a set of entities E if for every entity $e' \neq e$, e is distinguishable from e'.*

Distinguishable entities are the only one that can be designated by a definite description.

Definition 1 seems rather complicated (due to the universal quantifier in the second part) and thus needs some justification. Let us see some examples:

An entity e which is a cat is 0-distinguishable from an entity e' which is a dog because $\mathcal{P}(e)$=$\{\mathsf{cat}_1\}$ is not included in $\mathcal{P}(e')$=$\{\mathsf{dog}_1\}$.

An entity e which is a cat and which eats b (a bird) is 1-distinguishable from an entity e' which is a cat and which eats m (a mouse). Actually, $\mathcal{P}(e) = \{\mathsf{cat}_1, \mathsf{eat}_1\}$ is included in $\mathcal{P}(e') =$

$\{\mathsf{cat}_1, \mathsf{eat}_1\}$, but there exists an entity (b) with which e is in relation (via eat_1) and which is distinguishable from m, which is the only entity with which e' is in relation via eat_1. So, the situation can be resumed as in figure 1:

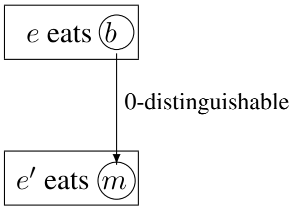

figure 1: e is 1-distinguishable from e'

If we add the information that e' also eats f (a fish), the conclusion remains true, as we can see on figure 2.

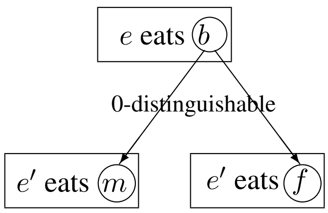

figure 2: e is 1-distinguishable from e'

But if we add the information that e' also eats b', a bird not distinguishable from b, then the conclusion is no longer true (see fig. 3).

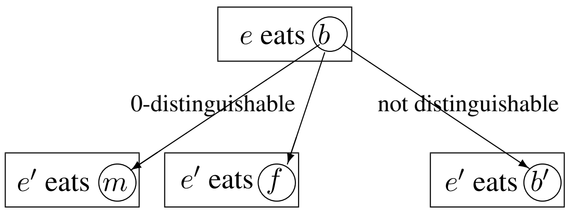

figure 3: e is not distinguishable from e'

e is not distinguishable from e', no definite description can designate e and not e'. So, we see that, in order for e to be distinguishable from e', b has to be distinguishable from all the entities which are in relation with e' via eat_1. That illustrates the necessity of the universal quantifier in definition 1.

Let us see a more complicated example, where tuples are involved.

$E = \{e, e', x_1, y_1, z_1, x_2, y_2, z_2\}$

e, e': man

x_1, z_1: ball – y_1 : cake

x_2, y_2: blond, child – z_2: child

e gives x_1 to x_2 (e gives a ball to a blond child)

e' gives y_1 to y_2 (e' gives a cake to a blond child)

e' gives z_1 to z_2 (e' gives a ball to a child)

The question is: Is e distinguishable from e'? The answer is clearly yes, "the man who gives a ball to a blond child" is a definite description that designates e and not e'.

First, e is not 0-distinguishable from e' ($\mathcal{P}(e) = \{\mathsf{man}_1, \mathsf{give}_1\}$ is included in $\mathcal{P}(e') = \{\mathsf{man}_1, \mathsf{give}_1\}$).

So, e is 1-distinguishable from e' if we find a relation R in $\mathcal{P}(e)$ and a tuple T that matches R with e and such that for each tuple T' that matches R with e', T' contains an entity e'_i from which the entity e_i in T is 0-distinguishable.

Let us check if this is true for give_1 and (x_1, x_2). $T_1 = (x_1, x_2)$ matches give_1 with e ($\mathsf{give}(e, y_1, z_1)$ is true). There are two tuples $T_2 = (y_1, y_2)$ and $T_3 = (z_1, z_2)$ that match give_1 with e'.
x_1 is 0-distinguishable from y_1. So it is right for T_2.
x_2 is 0-distinguishable from z_2. So it is right for T_3.

The situation can be resumed by the schema in figure 4:

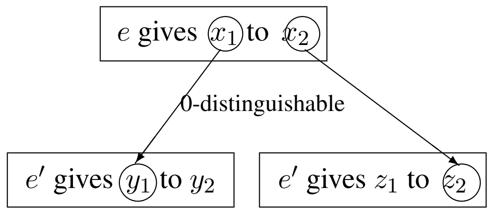

figure 4: e is 1-distinguishable from e'

Let us add "e' gives z_1 to y_2" to the above example:
$T_4 = (z_1, y_2)$ matches give_1 with e'. But x_1 is not distinguishable from z_1 and x_2 is not distinguishable from y_2. This new information prevents e being distinguishable from e'.
This case is represented on figure 5:

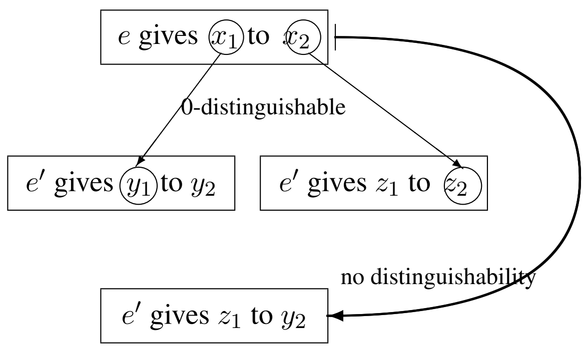

figure 5: e is not distinguishable from e'

Again, we see that it is not sufficient to check the existence of a tuple and a relation in $\mathcal{P}(e')$ that introduce the distinguishability to e via give_1. We have to check this for each tuple that matches give_1 with e'.

Moreover, one can also notice in the above example that the entity which "leads to" the k'-distinguishability is not unique. It may be different upon each tuple (x_1 for T_2 and x_2 for T_3). This is quiet different from the often used shortcut: e_1 is k-distinguishable from e_2 if it is in relation with *one* entity e'_1 which is k'-distinguishable from an entity e'_2 which is related to e_2.

So, although our definition may seem complicated, it cannot be simplified if we want it to seize the notion of distinguishability. We will now study some of its properties.

3 Some properties

This definition of the k-distinguishability of an entity leads to two interesting ideas:

- A set of entities can be organised in subsets or classes via a related notion, confusability. Confusability is a transitive relation and thus it defines a partial order on subsets of E.

- A notion of distance can be defined from k-distinguishability. Actually, the greatest k is, the less distinguishable the related entities are. The inverse of this k defines a distance between entities.

3.1 A partial order on the set of entities

Definition 3 *(Confusability C):*
We say that e_1 *is* ***k-confusable*** *with* e_2 *(we denote it* $e_1\, C_k\, e_2$*) when not* $e_1\, D_k\, e_2$.
We say that an entity e_1 *is* ***confusable*** *with another entity* e_2 *if* $e_1\, C_k\, e_2$ *for every* k *(we denote it* $e_1\, C\, e_2$*). It is equivalent to say that an entity* e_1 *is confusable with an entity* e_2 *if* e_1 *is not distinguishable from* e_2.

For example, e_1 is 1-confusable with e_2 if e_1 is not 1-distinguishable (nor 0-distinguishable) from e_2. But, in the same time, e_1 can be 2-distinguishable from e_2 and thus, not confusable with e_2.
We remark that if $e_1\, C_k\, e_2$, then $e_1\, C_j\, e_2$, for every $j < k$.

Intuitively, one would like C to be transitive (if an entity e_1 is confusable with an entity e_2 which is confusable with an entity e_3, then e_1 should be confusable with e_3).

Theorem 1 *C is transitive.*

Proof: We shall prove by induction on k that if $e_1\, C\, e_2$ and $e_2\, C\, e_3$, then $e_1\, C_k\, e_3$, for every $k \geq 0$.
If $e_1\, C\, e_2$ and $e_2\, C\, e_3$, then $\mathcal{P}(e_1) \subset \mathcal{P}(e_2) \subset \mathcal{P}(e_3)$, and so, $e_1\, C_0\, e_3$.

Let us suppose that, for every e_1, e_2 and e_3, if $e_1\, C\, e_2$ and $e_2\, C\, e_3$, then $e_1\, C_k\, e_3$, and that there exist three entities f, g, and h such that:
$f\, C\, g$, $g\, C\, h$ and $f\, D_{k+1}\, h$.
By the induction hypothesis, $f\, C_k\, h$, and so $\mathcal{P}(f) \subset \mathcal{P}(h)$. Thus, as $f\, D_{k+1}\, h$, there exist $(x_1, \ldots, x_n)$ and a relation R such that:
$R(f, x_1, \ldots x_n)$
$\forall (z_1, \ldots z_n)$ such that $R(h, z_1, \ldots, z_n)$, $\exists i \leq n, k' < k$ such that $x_i\, D_{k'}\, z_i$. (a)
(We have supposed, with no loss of generality, that f has rank 1 in R)
As $f\, C\, g$, $\exists (y_1, \ldots, y_n)$ such that:
$R(g, y_1, \ldots, y_n)$
$\forall i \leq n, x_i\, C\, y_i$
As $g\, C\, h$, $\exists (z'_1, \ldots, z'_n)$ such that:
$R(h, z'_1, \ldots, z'_n)$
$\forall i \leq n, y_i\, C\, z'_i$
Thus, for every $i \leq n$:
$x_i\, C\, y_i$ and $y_i\, C\, z'_i$
By the induction hypothesis, for every $i \leq n$, $x_i\, C_k\, z'_i$, which is in contradiction with (a).
□

We remark that C is reflexive and not symmetric. But, since C is a transitive relation, the relation $\mathcal{E}$ defined by $e_1\, \mathcal{E}\, e_2$ if $e_1\, C\, e_2$ and $e_2\, C\, e_1$ is an equivalence relation (with this relation, we put in the same class entities which are confusable) and C, when restricted to the quotient set (the set of the equivalence classes) $E/\mathcal{E}$, is a partial order that we denote $<_C$.

Since $<_C$ is an (partial) order relation on $E/\mathcal{E}$, which is a finite set, it has maximal and minimal elements. The maximal elements can be seen as *very well defined entities* (they are confusable with no other entity in other subsets) and the minimal elements as the *conceptual entities* (no entities in other subsets are confusable with them, but they are confusable with many other entities). We remark that two minimal entities (as two maximal ones) are not confusable, since the set of the minimal elements of an ordered set is an antichain (as the set of the maximal elements).

Thus, for example, a set of entities can be organised as in figure 6:

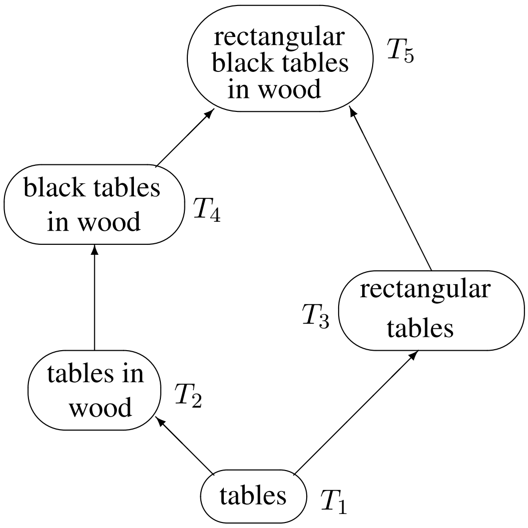

Figure 6: sets of entities ordered by $<_C$

$T_1 <_C T_2 <_C T_4 <_C T_5$
$T_1 <_C T_3 <_C T_5$

3.2 A distance between entities

Now, let us see that the notion of k-distinguishability leads to a notion of distance between entities. By now, if we take the smallest k such that e_1 is k-distinguishable from e_2 (we note it $\kappa(e_1, e_2)$ (if $e_1\, C\, e_2$, $\kappa(e_1, e_2) = \infty$)) the smaller $\kappa(e_1, e_2)$ is, the further e_1 is from e_2.

For example, if e_1 is 0-distinguishable from e_2, e_1 is very different from e_2 (a cat and a dog, for instance). But if e_1 is not 0-distinguishable from e_2 but is 1-distinguishable from it, then e_1 is nearer from e_2 (two cats, one that eats a bird and the other that eats a mouse, for instance).

So, one could expect that κ is like the inverse of a distance. Let us see that point.

Definition 4 *Let E be a set of entities. We define on $E/\mathcal{E}$:*

$\Theta(e, e) = 0$

$\Theta(e_1, e_2) = \max\{(\kappa(e_1, e_2) + 1)^{-1}, (\kappa(e_2, e_1) + 1)^{-1}\}$ *if* $e_1 \neq e_2$[2].

Theorem 2 Θ *is a distance on $E/\mathcal{E}$.*

We recall that a *distance* on a set X is an application $d : X \times X \rightarrow \mathbb{R}^+$ such that:

$\forall x, y, d(x, y) = 0 \iff x = y$
$\forall x, y, d(x, y) = d(y, x)$
$\forall x, y, z, d(x, y) \leq d(x, z) + d(z, y)$.

Theorem 2 follows immediately from the following:

Lemma 1 *If $e_1\, D_k\, e_2$, then, for every e_3:*
$e_1\, D_k\, e_3$ *or* $e_3\, D_k\, e_2$.

[2] We take $1/\infty = 0$

Proof of Lemma 1: The proof is by induction on k.
If $k = 0$, then $\mathcal{P}(e_1) \not\subset \mathcal{P}(e_2)$. Thus, if $\mathcal{P}(e_1) \subset \mathcal{P}(e_3)$ (i.e. $e_1\, C_0\, e_3$), then $\mathcal{P}(e_3) \not\subset \mathcal{P}(e_2)$, and so $e_3\, D_0\, e_2$.
Let us suppose that the property is true for $k - 1$ and that $\kappa(e_1, e_2) = k > 0$. There exists a relation R and $(x_1, \ldots x_n)$ with $R(e_1, x_1, \ldots, x_n)$ such that for every $(y_1, \ldots, y_n)$ with $R(e_2, y_1, \ldots, y_n)$ (such a $(y_1, \ldots y_n)$ exists, otherwise $\kappa(e_1, e_2) = 0$), there exists i with $x_i\, D_{k-1}\, y_i$.
(We have supposed, with no loss of generality, that e_1 has rank 1 in R)
Let $(z_1, \ldots, z_n)$ be such that $R(e_3, z_1, \ldots, z_n)$. If such a $(z_1, \ldots, z_n)$ does not exist, we would have $e_1\, D_0\, e_3$, and the property would hold for k. By the induction hypothesis, we have:
(a) $x_i\, D_{k-1}\, z_i$ or (b) $z_i\, D_{k-1}\, y_i$.
If there exists a $(z_1, \ldots, z_n)$ such that, for every $(y_1, \ldots, y_n)$, we are in case (b), then $e_3\, D_k\, e_2$.
Otherwise, for every $(z_1, \ldots, z_n)$ such that $R(e_3, z_1, \ldots, z_n)$, there exists a $(y_1, \ldots, y_n)$ for which we are in case (a). In fact, $(y_1, \ldots, y_n)$ does not matter for this case, and so, that is to say that $e_1\, D_k\, e_3$.
□

Actually, this lemma shows much more than theorem 2. It says that the entity set is structured by distinguishability in such a way that whatever the couple of entities we take, there is no other entity between them. This lemma induces a stronger property for Θ:
Let d be a distance on a set X. If we have:
$\forall x, y, z, \max\{d(x, y), d(x, z)\} \geq d(z, y)$
(which is equivalent to say that for any triple, the two greatest distances are equal[3]), then the distance is *ultrametric*.

Theorem 3 Θ *is an ultrametric distance on $E/\mathcal{E}$.*

Ultrametric distances have a lot of properties (See (Barthélémy and Guénoche, 1991)). In particular, they are equivalent to a hierarchical classification of the underlying set[4] (like the phylogenetic classification of natural species).

More precisely, given a set X with an ultrametric distance d, the sets $C_{x,y} = \{z/d(x, z) \leq$

[3] Suppose that for a triple (x, y, z), we have, for instance, $d(x, y) \geq d(x, z) \geq d(y, z)$. Since $\max\{d(y, z), d(x, z)\} \geq d(x, y)$, we also have $d(x, z) \geq d(x, y)$, and thus $d(x, z) = d(x, y)$.

[4] The set is partioned into non-overlapping subsets, each subset being (eventually) divided into non overlapping subsets,. . .

$d(x,y)\}$ form a hierarchical classification of X. Conversely, given a finite set X with a hierarchical classification, if, for $x \neq y$, we define $d(x,y)$ as the cardinality of the smallest class containing x and y, and $d(x,x) = 0$ for all x in X, then d is an ultrametric distance.

In addition, given a set X with an ultrametric distance d, there exists a tree (called *ultrametric tree*) with labels on its internal nodes, its leaves indexed by the elements of X and such that:

- for any two leaves x and y, the label of their lowest common ancestor is $d(x,y)$.
- for any leaf x, the labels on the path from the root to x form a decreasing sequence.

For instance, with the example shown on figure 5, we obtain the tree on $E/\mathcal{E}$ which is shown on figure 7 (for this example, since there is no pairwise confusable entities, $E/\mathcal{E} = E$):

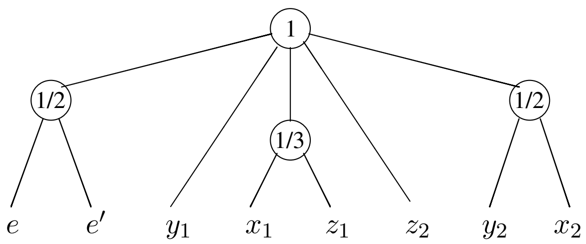

Figure 7: a tree on $E/\mathcal{E}$

On this tree, given a couple of entities, one can see the difficulty to distinguish them. This information has been construct in a global way (by using all the relations between entities) and it is rather different (and more accurate) from what one would say at a first glance. For instance, we can see that x_1 and y_1 are more difficult to distinguish than x_2 and z_2 or than e and e' (the label of their lowest common ancestor is 1/3 instead of 1/2).

4 An algorithm for searching distinguishable entities

The algorithm is based on dynamic programming (Aho et al., 1974). This is a standard technique which is used, for instance, to calculate distances in graphs. We work on a set $E = \{e_1, \ldots e_n\}$ of entities. The main structure is a $n \times n$ matrix $\mathcal{M}$. At each step k, the algorithm determines the couples (e_i, e_j) of entities such that $\kappa(e_i, e_j) = k$ and loads k into $\mathcal{M}[i,j]$.

- At step 0, we check for each couple (e_i, e_j) whether $\mathcal{P}(e_i) \subset \mathcal{P}(e_j)$ or not. If $\mathcal{P}(e_i) \not\subset \mathcal{P}(e_j)$, we load 0 into $\mathcal{M}[i,j]$.
- At step $k > 0$, for every couple (e_i, e_j) such that $\mathcal{M}[i,j]$ is not yet calculated, we determine if $\kappa(e_i, e_j) = k$ or not, using already calculated values in $\mathcal{M}$ to check conditions of definition 1. If it is the case, we load k into $\mathcal{M}[i,j]$.
  If no value of $\mathcal{M}$ is updated, then the algorithm stops (if there are no e, e' in E such that $e\, D_k\, e'$, then there exist no f, f' in E such that $f\, D_{k+1}\, f'$)

At the end of the algorithm, if $e_i\, D\, e_j$, $\mathcal{M}[i,j]$ contains $\kappa(e_i, e_j)$. We also compute an auxiliary matrix $\mathcal{A}$ in which we put the relations that have been used to calculate $\kappa(e_i, e_j)$. The matrix $\mathcal{A}$ will be used to build referring expressions.

The algorithm runs in $O(n^2 \cdot K \cdot N \cdot T^2)$, where $K = \max\{\kappa(e,e'), e\, D\, e'\}$, N is the greatest property arity, and T is the cardinality of the greatest set $\mathcal{T}(e_i)$ of all couples (p,t), where p is a property and t a tuple that matches p with e_i.

N, T and K are rather small and can be assimilated to constants[5]; so, if we are only concerned with the number of entities, our algorithm is in $O(n^2)$.

Let us see how it works on an example from (Croitoru and van Deemter, 2007):

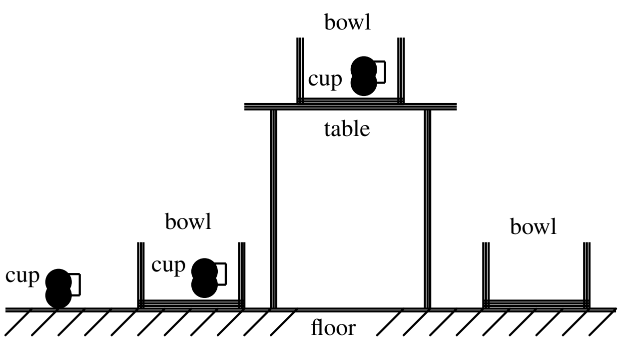

Figure 8: a scene

Croitoru and van Deemter (2007) represent the scene of figure 8 by an entity set $E = \{v_0, \ldots v_7\}$ with the following properties:

v_0, v_3, v_7: cup
v_1, v_5, v_6: bowl
v_2: table

[5]Actually, from a theoretical point of vue, we only have $K \leq n$, and no limit on T and N. But, from a practical point of vue, one can have a scene with (for instance) 10000 entities, but there is no property of arity 10, no entity with 100 properties and no distinguishing expression of length 50 (even if such an expression would exist, it would be impossible to use it); so N, T and K are small

v_4: floor
v_0 is in v_1
v_1 is on v_2
v_3 is on v_4
v_2 is on v_4
v_5 is on v_4
v_6 is on v_4
v_7 is in v_6

Our algorithm produces the following matrix $\mathcal{M}$ (due to lack of space, we do not show the matrix $\mathcal{A}$: its breadth would exceed the sheet):

$$\mathcal{M} = \begin{array}{c} \\ v_0 \\ v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \\ v_6 \\ v_7 \end{array} \begin{array}{c} \begin{array}{cccccccc} v_0 & v_1 & v_2 & v_3 & v_4 & v_5 & v_6 & v_7 \end{array} \\ \begin{pmatrix} / & 0 & 0 & 0 & 0 & 0 & 0 & 2 \\ 0 & / & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & / & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & / & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & / & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & / & \infty & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & / & 0 \\ 2 & 0 & 0 & 0 & 0 & 0 & 0 & / \end{pmatrix} \end{array}$$

With this matrix $\mathcal{M}$, one can easily determine which entities are distinguishable: they are the one with no $+\infty$ on their line. Here, we can see that v_5 is not distinguishable: it is distinghishable from all entities but v_6

It is also easy to construct sets of distinguishing properties, using matrix $\mathcal{A}$. For instance, if we want to distinguish v_0 from v_7, we use the following elements of $\mathcal{A}$:

$\mathcal{A}[v_0, v_7] = \{(isin_1, 2, v_1, v_6)\}$
$\mathcal{A}[v_1, v_6] = \{(ison_1, 2, v_2, v_4)\}$
$\mathcal{A}[v_2, v_4] = \{table_1, ison_1\}$.

Since v_2 is 0-distinguishable from v_4, we get the following distinguishing formula:

$\lambda x\, \lambda y\, \lambda z\ isin(x, y) \wedge ison(y, z) \wedge table(z)$[6]

from which one can easily obtain the following expression which distinguishes v_0 from v_7: "*the entity which is in an entity which is on an entity which is a table*".

Using this method, we obtain minimal expressions to distinguish one entity e from another entity e'. A referring expression (which distinguishes one entity e from all the others) can be obtained by computing the conjunction of all these minimal expressions. This conjunction contains many redundancies, and it can be reduced in $O(n \log n)$. Actually, by this way, one generally obtains an expression which is very close to the expression which distinguishishes e from the nearest other entity (i.e. the entity e' for which $\kappa(e, e')$ is maximal). For instance, in the example above, the expression which distinguishes v_0 from v_7 is a referring one for v_0: there is no other entity "*in something on a table*".

So, we get sets of distinguishing properties for all the distinguishable entities of a scene in polynomial time (and more precisely in $O(n^2 \log n)$). This is much better than the methods of Kramer and al. (2003) and of Croitoru and van Deemter (2007), which both rely on subgraph isomorphisms (which is a NP-complete problem).

[6]We can obtain another distinguishing expression by taking $ison_1$ instead of $table_1$ in $\mathcal{A}[v_2, v_4]$. We choose $table_1$ because its arity is smaller, so we get a simpler formula.

5 Conclusion

The two main results of this paper are:

- An efficient algorithm to compute distinguishing descriptions. Our algorithm is efficient enough to be applied on complex scenes.
- An ultrametric distance which captures the difficulty to distinguish two entities and provides a phylogenic classification of the entities.

These two results follow from our definition of k-distinguishability. More precisely, they are due to the incremental nature of the k-distinguishability, which thus reveals to be a pivot for the Generation of Referring Expressions (GRE).

Acknowledgments

P. Préa is supported in part by ANR grant BLAN06-1-138894 (projet OPTICOMB)

References

Alfred V. Aho, John E. Hopcroft, and Jeffrey D. Ullman. 1974. *The Design and Analysis of Computer Algorithms*. Addison-Wesley, Reading, MA.

Jean-Pierre Barthélémy and Alain Guénoche. 1991. *Trees and Proximity Representations*. J. Wiley & sons, New York, NY.

Madalina Croitoru and Kees van Deemter. 2007. A conceptual graph approach to the generation of referring expressions. *International Joint Conference on Artificial Intelligence*, Hyderabad.

Robert Dale. 1989. Cooking up referring expressions. *Proceedings of the Twenty-Seventh Annual Meeting of the Association for Computational Linguistics*, Vancouver.

Robert Dale and Nicholas Haddock. 1991. Generating Referring Expression Involving Relations. *Proceedings of the fifth conference of the European ACL*, Berlin.

Robert Dale and Ehud Reiter. 1995. Computational Interpretations of the Gricean Maxims in the Generation of Referring Expressions. *Cognitive Science*, 19(2):233-263.

Kees van Deemter. 2002. Generating Referring Expressions: Boolean Extensions of the Incremental Algorithm. *Computational Linguistics*, 28(1):37-52.

Claire Gardent. 2002. Generating Minimal Definite Descriptions. *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, Philadelphia.

Albert Gatt and Kees van Deemter. 2006. Conceptual Coherence in the Generation of Referring Expressions. *Proceedings of ACL*, Sydney.

Helmut Horacek. 2003. A Best-First Search Algorithm for Generating Referring Expressions. *Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics*, Budapest.

Emiel Krahmer, Sebastian van Erk and André Verleg. 2003. Graph-based Generation of Referring Expressions. *Computational Linguistics*, 29(1):53-72.

Jette Viethen and Robert Dale. 2006. Algorithms for Generating Referring Expressions: Do They Do What People Do? *Proceedings of the International Conference on Natural Language Generation*, Sydney.