Relational n-gram databases as a basis for unlimited annotation on large corpora

Mark Davies
Illinois State University

1 Introduction

One of the fundamental problems facing the creators of large corpora is the need to balance size, speed, and an advanced search engine that allows a wide range of queries. Some approaches may achieve two of the three goals quite well, but weaknesses in the third goal severely limit the usability of the corpus. For example, one approach might place a number of levels of annotation within the textual corpus itself. Yet if it is necessary to traverse the entire corpus in order to carry out each search, then this would probably result in a corpus that is unusable for all but the most dedicated of researchers. Another approach might use an "off the shelf" product like Microsoft Search to index and search the corpus. This will result in very fast queries (one or two seconds to search a 100 million word corpus), but is limited in that it allows searches mainly for exact words and phrases, with little or no possibility of traditional annotation like lemma and parts-of-speech.

Within the past decade – and particularly the past five years – a number of "mega-corpora" have come online, which have attempted to address these problems. These include the BNC (Clear 1993, Berglund 1999), Cobuild (Sinclair 1987, Clear et al 1996), ARTFL (Olsen and Hinkelman 1991), CETEMPúblico (Rocha and Santos 2000, Santos and Bick 2000), and CORDE/CREA (http://www.rae.es). (For a general introduction to this problem of competing goals in the creation of large corpora, see also McEnery and Wilson 1996, Kennedy 1998, and Biber et al 1998).

In this paper we will consider an alternative approach which is unique in the way that it addresses the competing need for speed, size, and searchability. This approach is based on the use of n-grams that are stored in large relational databases, and a finished example of this approach can be found in the 100 million word, NEH-funded "Corpus del Español" (www.corpusdelespanol.org) that was completed in 2002. This approach allows for corpora of essentially unlimited size, very fast retrieval (usually one or two seconds), and a very wide range of annotation on the corpus – such as parts-of-speech, lemma, synonyms, frequency, and customized lists. In addition, its modular design means that any number of additional annotation features can be added in the future, with absolutely no decrease in performance.

2 Overview of the corpus architecture

In terms of the overall design of the corpus, the approach is quite unique in that there is very little annotation on the textual corpus itself. The 100 million word corpus is stored as 1000-2000 word chunks of text in a Microsoft SQL Server 7.0 database, but this textual corpus itself is not annotated in any way, apart from a code that indicates the source of each block of text. However, it is indexed with SQL Server "Full-Text Indexing", which is similar to the standard Microsoft Search engine. This indexing scheme allows exact words and phrases to be found quickly – usually less than one second to query the entire corpus and return the relevant examples. The important limitation, however, is that the Full-Text search engine for SQL Server only works well with exact words and phrases. Even wildcard searches are problematic, and certainly there is no capability for customized annotation of any sort.

The annotation for the Corpus del Español resides not in the textual database, but rather in separate relational databases that contain tables of all of the distinct 1, 2, 3, and 4-grams in the corpus. These tables also include the frequency for each of these n-grams in each of the centuries from the 1200s-1900s, as well as the different registers of Modern Spanish. As might be imagined, the tables are rather large. There are nearly one million distinct 1-grams (i.e types), eleven million distinct 2-grams, forty million distinct 3-grams, and 65 million distinct 4-grams. An example of one of the forty million 3-grams from the corpus is the following:

Table 1. N-grams/frequency table in SQL Server

w1	w2	w3	x12	x13	x14	x15	x16	x17	x18	x19	19-Lit	19-Oral	19-Misc
hora	en	que	31	0	8	61	55	39	369	92	78	10	4

The columns w1, w2, w3 refer to each of the "slots" in the 3-gram; the columns x12-x19 refer to the frequency of this 3-gram in the 1200s-1900s; and 19-Lit, 19-Oral, and 19-Misc refer to the frequency in these three registers from the 1900s. Each of these relational database tables is indexed, including some clustered indices (to be discussed in more detail later on), all of which leads to very fast retrieval.

3 Simple pattern-matching queries

Even without annotation, there are a number of useful queries that can be done directly on the n-gram/frequency databases. For example, a user can input the following string into two fields in the web-based form and the script will convert this into the SQL query:

[SEARCH] *aren [SORT] 1200s [LIMITS] +1200s +1400s –1900s

```
select * from x1 where
w1 like ('%aren') and
x12>0 AND x14>0 AND x19= 0
order by x12 desc
```

This will search the [1-grams] table for all cases where the word [w1] ends in [-aren] (a marker for the archaic 3PL future subjunctive), which occur at least five times in the 1200s and the 1400s, but which have disappeared by the 1900s, and then sort the results by the frequency in the 1200s. This will result in cases like *fallaren* "that they might lack" and *tomaren* "that they might take".

A second example is the following web-based query, with the resulting SQL query:

[SEARCH] tan * como [SORT] 1900s

```
select * from x3 where
w1 in ('tan') and
w3 in ('como')
order by x19 desc
```

This query will search the [3-grams] table for all cases where the [w1] column is *tan* "as/so" and the [w3] column is *como* "as", and order the results by the frequency in the [x19] column. This will produce hits like *tan bueno como* "as good as", *tan rápido como* "as fast as", etc.
The final query is somewhat more complex:

[SEARCH] me/te/nos/le/les quier* *r [SORT] 1900s

```
select * from where
w1 in ('me', 'te', 'nos', 'le', 'les') and
w2 like ('quier%') and
w3 like ('%r')
order by x19 desc
```

This query will search the [3-grams] table for all records where the first word [w1] is one of the following [*me,te,nos,le,les*] (indirect object pronouns), the second word [w2] has the pattern [*quier-*] (some forms of the verb *querer* "to want"), and the third word [w3] ends in [*-r*] (possibly an infinitive). This will produce strings like *le quiero decir* "I want to tell him" or *nos quiere llamar* "wants-3SG to call us"
After submitting the query, users see a list of all matching strings, along with the frequency of each string in each of the centuries from the 1200s to the 1900s, as well as three different registers of Modern Spanish (spoken, written-fiction, and written-non-fiction). An example of the output (from the third query shown above) is found in the following table:

Table 2: N-grams / frequency – web page results

#	PHRASE(S)	12	13	14	15	16	17	18	19	Lit	Oral	Misc
1	te quiero decir				17	10	1	8	49	11	38	
4	me quiero ir				32	10	1	2	23	7	16	
19	le quiere dar	9	1	1	7	6	2	6	4		3	1
22	te quiero contar			1	11	3			4		4	
	. . .											

Many users will only want to see a list of the matching words and phrases, with their frequency in each period. However, there are also many different options to select certain words and phrases, in order to see them in KWIC format. Users can click on a phrase to see it in all historical periods, or they can click on a number in one of the columns to see one word or phrase in that historical period. Alternatively, they can select multiple words and phrases and multiple historical periods (or registers of Modern Spanish) and see the KWIC display for just those selected items.

As with a traditional KWIC display, users can re-sort the occurrences by left or right context words, and can also select certain phrases to see in more detail. A sample of the KWIC display is seen in the following table (the entries have been shortened in the interest of space):

Table 3: KWIC display on the web

1/1	CENTURY	TEXT	RE-SORT BY: L-2 L-1	C	R-1 R-2
3	12	Libro de los fuero..	pertenesca & tiene gela forçada. Et non	le quiere dar	lo que a tomado & en logar de dar gelo
8	15	La Serrana de la V..	ha sido mi desdicha el desengaño. No	me quiero casar,	padre, que creo que mientras no me caso
14	19_LIT	Follaje en los ojo..	siempre! ¡Haré lo que quiera, no	me quiero ir!	Ya soy grande y sé hacer de todo: he
27	19_ORAL	España Oral: CCON0..	achuchan a mi madre y a mi padre -	Te quiero decir	que es una cosa que yo - y mis padres
	. . .	. . .	. . .	. . .	. . .

Because Spanish has morphology that is both strong and fairly regular, users can employ simple lists of words and word patterns to search for even relatively complex syntactic constructions, as in Table 2 above. As a result of the indexes on the n-grams tables in the relational database, these searches are also very quick – nearly always just one or two seconds for even the most complex searches. However, at some point it will obviously be necessary to have more complete annotation, including annotation for those lemma that are not morphologically regular (e.g. *quis** for preterite forms of *querer*), as well as parts of speech that are not predictable in terms of forms (such as nouns and adjectives in Spanish). In the following section we discuss how that can be carried out, using relational databases.

4 Alternate approaches to the annotation architecture

There are two possible approaches to the question of where and how to place POS and lemma information in the relational database. One option is to place it within the same tables that contain the n-grams and frequency information. The other is to separate it from these tables and use SQL JOIN statements to join together the annotation and n-grams/frequency tables. In the following section, we will discuss the relative advantages and disadvantages of both of these approaches.

In the first approach, then, we would include the POS and lemma information as additional fields of the n-grams/frequency tables. For example, in the following case there is annotation (POS and lemma) within the table for each of the three words in the 3-gram

Table 4: N-grams/frequency table with integrated POS and lemma annotation

W1	L1	C1	W2	L2	C2	W3	L3	C3	x12	x13	x14	x15	x16	x17	x18	x19	19-Lit	19-Oral	19-Misc
no	no	adv	puede	poder	v_pres	ser	ser	v_inf	105	13	140	518	423	269	892	646	200	339	107

In this schema, a user who is searching for cases of a form of *no* + form of *poder* + infinitive (e.g. *no puede ser* "can-not-3SG be" or *no podemos decir* "we cannot say") would enter the following in the web-based form (note that the period+asterisk after a word signifies lemma, and an asterisk+period before a word signifies part-of-speech):

no poder.* *.v_inf

This search phrase is then translated into the following SQL command, which queries only the 3-grams table [x3], since it has all of the POS and lemma information included within that one table:

```
select top 300 * from x3 where
        W1 = 'no' and
        L2 = 'poder' and
        C3 = 'v_inf'
```

The advantage of having the annotation within the n-grams / frequency table itself is that the contextual information can be used to resolve ambiguity. For example, in the 3-grams table two of the 40+ million records are for the strings *no puede ser* "can-not-3SG be" and *de un ser* "of a (human) being":

Table 5. Contextual disambiguation, based on n-grams

W1	L1	C1	W2	L2	C2	W3	L3	C3	x12	x13	x14	x15	x16	x17	x18	x19	19-Lit	19-Oral	19-Misc
no	no	adv	puede	poder	v_pres	ser			105	13	140	518	423	269	892	646	200	339	107
de	de	prep	un	un	a_ind	ser			0	0	2	11	5	3	13	37	6	17	14

In Spanish, *ser* can either be a noun (*el ser humano* "the human being") or an infinitive (*para ser bueno* "in order to be good"). Therefore, it is not immediately clear what the POS or the lemma should be for the [*ser*] in [w3] slots.

Using simple SQL updates, however, we can look at the contextual information to fill in or alter the POS and lemma values, based on the values in the other word slots. For example, the following UPDATE query will assign a value of [v_inf] when [*ser*] is preceded by a modal verb such as *poder* (the first row), and it will assign the value of [n] to [*ser*] when it occurs after an indefinite article (as in the second row). The first case is shown in the following example:

```
update x3
set c3 = 'v_inf'
where L2 = 'poder' and w3 = 'ser'
```

Researchers who are used to working with a language such as English – which has relatively poor morphology – might object to the idea of having the POS and lemma information in tables where they are removed from their original context – which in our case is the unannotated textual corpus that is stored in 2000 word chunks in a separate database. The concern might be that there would not be enough information in the n-grams database to successfully disambiguate cases of polysemy. For a language such as Spanish, however, this concern is probably unfounded. Because Spanish is such a morphologically strong language, there are relatively few forms (as with *ser*) that have a high frequency as two different parts of speech or two different lemma, and which cannot be successfully disambiguated with even a very limited context..

In fact, for a language like Spanish, an alternate approach is to remove the annotation even further from the original context and have the annotation tables separate from even the n-grams tables, which shows the immediate context of the word. The following tables show this architecture, in which the n-grams/frequency information is in one table, and the POS and lemma information are in two additional tables:

Table 6: Separate n-grams/frequency, POS, and lemma tables

W1	W2	W3	x12	x13	x14	x15	x16	x17	x18	x19	19-Lit	19-Oral	19-Misc
no	puede	ser	105	13	140	518	423	269	892	646	200	339	107

x3

w1	x1
no	adv
puede	v_pres
ser	v_inf
ser	n

pos (x_c)

w1	x1
no	no
puede	poder
ser	ser

lemma (x_L)

In this scenario, there would be simple SQL JOIN commands to link the two databases. The same query shown above would be translated into the following SQL command. In this case, the database first sub-queries the POS (x_c) and lemma (x_L) tables, and then feeds the output from these tables into a query of the main n-gram/frequency table (x3):

```
select top 300 * from x3 where
        w1 in ('no') and
        w2 in (select w1 from x_L where x1 in ('poder')) and
w3 in (select w1 from x_c where x1 in ('v_inf'))
```

Because of the very low level of polysemy in Spanish, and because the immediate context can usually be used successfully for disambiguation , there are relatively few cases where the non-contextual annotation presents a problem. For example, if a user searches for [no poder.* *.v_inf], then [*no podia estar*], [*no pudiéramos contar*], [*no puede ser*] and others will be retrieved, because *ser* is listed as a [v_inf] in the POS table. And it turns out that all of these examples of the potentially polysemous *ser* really are in its use as an infinitive – there would be only a very small percentage of cases where *ser* is the noun "(human) being". Likewise, if a user searches for [el *.n humano], then [*el ser humano*] "the human being" will be retrieved, and again virtually all of these cases will

be with *ser* as a noun (because of the preceding determiner and the following adjective). The only problem in the assignment of part-of-speech and lemma will occur in those relatively few cases in which 1) a word is polysemous and 2) both meanings are highly frequent and 3) the context is not sufficiently rich to disambiguate the multiple meanings. Our experience from more than a year's worth of working with the Corpus del Español shows that there are in fact very few cases where serious ambiguity arises, and even in these cases the spurious phrases can simply be ignored by the end user.

One of the major advantages of placing the annotation in tables that are separate from the context to which they refer deals with redundancy. In the database architecture in which the POS and lemma information is part of the n-grams tables, then redundant annotation occurs every time a word occurs in any slot of any n-gram table. Thus a word like *es* "is" would be annotated as [POS=v_pres; lemma = ser] in each of the 209,160 rows of the 3-grams table where it appears in the [w2] slot – as well as hundreds of thousands of other rows for the other slots of the 3-grams table and the other n-grams tables. By placing the annotation in a separate table, there is exactly one entry for [*es*]. There are secondary benefits from placing the annotation in separate tables, both of which are related to the issue of redundancy. First, because the annotation only occurs in one or two rows of the POS or lemma tables, hundreds or thousands can be updated in a matter of one or two seconds. If a form can occur in hundreds of thousands of rows, on the other hand, then updating the annotation for the large amount of forms becomes more difficult.

A second issue is somewhat more technical in nature, and deals with the physical architecture of database tables. In most databases, only one column in each table can have a "clustered" index, which means that the rows in the table are physically arranged on the hard drive according the contents of that column. (In a non-clustered index, on the other hand, there are pointers to the data, but the data itself may be spread over the entire hard drive.) For our case, the important point is that if the clustered index is placed on the [w1] column (the first word in the n-gram), then any indices on the annotation columns in the n-grams table cannot be clustered, and queries dealing with POS or lemma will be relatively slow. By placing POS and lemma annotation in their own tables, each of these tables can contain a clustered index, and text retrieval will be much faster – typically just a second or two for even the most complex queries.

5 Simple part-of-speech and lemma queries

Now that we have discussed the annotation architecture, and the rationale for adopting our particular approach, let us briefly consider how end users are able to access this information from the web-based corpus. We will consider three different sample searches, which involve an increasing degree of complexity.

First, suppose that a user wants to know which are the most common verbs appearing in the phrase [difícil de VINF] "hard to V". The user enters the following into the web-based form, and this is converted by the script into the following SQL command:

[SEARCH] difícil.* de *.v_inf

```
select * from x3 where
w1 in (select w1 from x_L where x1 in ('difícil')) and
w2 in ('de') and
w3 in (select w1 from x_c where x1 in ('v_inf'))
  order by x19 desc
```

The sub-queries select the lemma (line 2) and part-of-speech (line 4) information for [difícil] and [v_inf] from lemma and part-of-speech tables. It then uses this information to search the actual 3-grams table [x3], and orders the results by the frequency of these n-grams in the 1900s (field x19), producing results like *difícil de explicar* "hard-SG to explain" (23 occurrences in the 1900s), *difíciles de encontrar* "hard-PL to find" (12 occurrences), and *difícil de creer* "hard-SG to believe" (12 occurrences).

Because the n-grams tables include information on the frequency of each of the n-grams in each of the sub-corpora, this can be used directly as part of the query syntax. For example, a user might want to know which adjectives are used with *situación* "situation" or *condición* "condition" at least two times in the 1900s, but which do not appear in the 1700s or 1800s. The user would input the following, and this would be converted to the SQL command:

[SEARCH] situación/condición *.adj [SORT] 1900s [LIMIT] 1900s>2 –1800s –1700s

```
select * from x2 where
```

```
w1 in ('situación', 'condición') and
w2 in (select w1 from x_c where x1 in ('adj')) and
x19>2 AND x18= 0
order by x19 desc
```

This will produce results like *situación concreta* "concrete situation", *situación incómoda* "uncomfortable situation", and *condición existencial* "existential condition".

Finally, it is posible to group the results by lemma. For example, user might want to know which verbs occur most commonly in the present subjunctive after the permissive verb *dejar* "to allow". They would enter the following into the search form (note the GROUP BY option}, and it will be converted to the SQL command:

```
[SEARCH] dejar.*  que *.v_subj_pres [GROUP BY]  lemma
```

```
select * from x3 where
w1 in (select w1 from x_L where x1 in ('dejar')) and
w2 in ('que') and
w3 in (select w1 from x_c where x1 in ('v_subj_pres'))
```

In this case, the script would then add another step, which would be to use a subsequent SQL GROUP command to group together all of the occurrences into the appropriate lemma. This would produce results like [*dejar que hacer*] "to allow someone to make", [*dejar que caer*] "to let something fall" (i.e. "to drop") and [*dejar que hablar*] "to let someone speak". Finally, we should note the speed of the queries. In all three of the preceding searches, it took less than two seconds to query the entire database and retrieve the matching strings.

6 Advanced comparisons using sub-queries

Perhaps the best example of the power of the relational database approach is the way in which these databases allow extremely complex comparisons of competing structures. For example, suppose that a user wanted to know which nouns occur more frequently with the adjective *blanco* "white" than with *negro* "black". With our approach, it would be quite easy to check for this. The user would enter the following into the search form on the web, and this would be converted to the following SQL command,:

```
[ SEARCH 1]  *.n blanco.* [LIMITS] 1900s > 10     [ GROUP BY ]  lemma
        [ SEARCH 1]  *.n negro.*        [LIMITS] 1900s < 5 [ GROUP BY ]  lemma
```

```
select blanco.freq as blanco,negro.freq as negro,blanco.w1 as w1 from
(
select top 10000 sum(x2.x19) as freq,x_l.x1 as w1
from x2,x_l where
x2.w1 in (select w1 from x_c where x1 in ('n')) and
x2.w2 in (select w1 from x_L where x1 in ('blanco')) and
x2.w1 = x_l.w1
group by x_l.x1
order by sum(x2.x19) desc
) blanco
left join
(
select top 10000 sum(x2.x19) as freq,x_l.x1 as w1
from x2,x_l where
x2.w1 in (select w1 from x_c where x1 in ('n')) and
x2.w2 in (select w1 from x_L where x1 in ('negro')) and
x2.w1 = x_l.w1
group by x_l.x1
order by sum(x2.x19) desc
) negro
on blanco.w1 = negro.w1
where blanco.freq > 10  and negro.freq < 5
order by blanco.freq desc
```

This SQL query uses two sub-queries to retrieve the 10,000 most frequent nouns with *blanco* and then with *negro*, and then limits the output to just those that occur more than ten times with *blanco* and less than five times with *negro*. The query takes less than five seconds, and returns a list including *vino* "wine", *guante* "glove", and

diente “tooth”. The opposite query – nouns that occur more frequently with *negro* – takes another four seconds and returns a list containing *agujero* “hole”, *humor* “(sense of) humor”, and *terciopelo* “velvet”.
Likewise, a user could quickly and easily check to see which verbs in Spanish appeared in the preterite much more often than they did in the imperfect. The user would enter the following into the search form on the web, and it would be converted to the following SQL query:

```
[ SEARCH 1]  *.v_pret  [LIMITS] 1900s > 200   [ GROUP BY ]  lemma
        [ SEARCH 2]  *.v_impf[LIMITS] 1900s < 40     [ GROUP BY ]  lemma
```

```
select pret.freq as pret,impf.freq as impf,pret.w1 as w1 from
(
select top 10000 sum(x1.x19) as freq,x_l.x1 as w1
from x_l,x_c,x1
where x_l.w1=x1.w1 and x_c.w1=x1.w1 and x_c.x1 = 'v_impf'
group by x_l.x1
order by sum(x1.x19) desc
) impf
left join
(
select top 10000 sum(x1.x19) as freq,x_l.x1 as w1
from x_l,x_c,x1
where x_l.w1=x1.w1 and x_c.w1=x1.w1 and x_c.x1 = 'v_pret'
group by x_l.x1
order by sum(x1.x19) desc
) pret
on impf.w1 = pret.w1
where pret.freq > 200 and impf.freq < 40
order by pret.freq desc
```

This SQL query uses two sub-queries to retrieve the 10,000 most frequent verbs in the imperfect and in the preterite, and then limits the output to just those that occur more than 200 times in the preterite and less than 40 times with the imperfect. The query takes less than five seconds and returns a list that includes the verbs *conquistar* “to conquer”, *fallecer* “to die”, and *aclarar* “to make clear”. The opposite search – verbs that are more frequent in the imperfect – also takes less than five seconds and yields verbs like *soler* “to be in the habit of Ving”, *lucir* “to shine”, and *carecer* “to lack”.

7 Using n-grams and frequency to annotate unknown forms

As we have seen, two of the main advantages of the n-grams / relational database approach are speed and the complexity and depth of the queries. However, there are two other important advantages as well. The first is the way in which the database architecture can be used to assist in the development of a lexicon for the corpus. The second advantage is the modular design, which allows an unlimited number of types of annotation, without any decrease in performance. We will address the issue of the lexicon in this section, and the issue of the modular architecture in the following section.

Turning first to the lexicon, when we created the Corpus del Español we also created a fairly robust lexicon, which contained POS and lemma information for more than 400,000 word types in the corpus. Even this lexicon was inadequate, however, because the farther back that one goes in the historical corpus, the lower the percentage of forms that can be annotated with the Modern Spanish lexicon. For example, the lexicon contained nearly 100% of the word types from the 1900s portion of the corpus, but this decreased to 40% from the 1700s, 33% in the 1500s, and 16% in the 1200s. In other words, most of the types from older historical periods were from a different “language”, as far as the Modern Spanish lexicon was concerned. We will see, however, that by using an approach based on n-grams tables in relational databases, we were still able to annotate tens of thousands of distinct word forms from the oldest stages of the language in a matter of just a few hours.

The n-grams tables provided valuable help in identifying the POS of word types that were not in the Modern Spanish lexicon. For example, suppose that we want to identify the 2000 most common nouns in the 1200s. A syntactic environment in which these would occur is [indef art] + __ + [que] (*un omne que* “a man that”, *vna casa que* “a house that”, etc). The following query – which takes less than one second to run – selects the 2000 most frequent words in slot 2 (w2) in the 3-grams table (x3), which occur more than two times in that slot and

which are preceded by a word (w1) that is (*una, una, vn, vna*) and which are are followed by *que* "that" in the third (w3) slot:

```
select top 2000 w2,sum(x12)
        from x3 where
        w1 in ('un', 'una','vn', 'vna') and
        w3 = 'que'
        group by w2
        having sum(x12) > 2
        order by sum(x12) desc
```

Likewise, the following query – which can be run against the entire 100 million word corpus in less than one second – is an attempt to define a syntactic environment for adjectives. It selects all words in the third slot (w3) of the 3-grams table (x3), in which the first slot (w1) is one of several high-frequency forms of *ser* "to be", and the second slot (w2) is a form of *muy* "very" or *tan* "so":

```
select w3,sum(x12)
        from x3 where
        w1 in ('es','era','será','sera','fue') and
        w2 in ('muy','mui','muy'','tan')
        group by w3
        having sum(x12) > 2
        order by sum(x12) desc
```

The preceding SQL queries select all word types that occur in a particular syntactic environment, and it is quite simple to go from here to the actual assignment of POS in these cases. First, we want to identify just those forms that do not already have a POS assigned. For example, *casa* "house" is already marked as [noun] in the Modern Spanish lexicon, and so there is no need to "re-annotate" this form for Old Spanish. But words like *omne* (= *hombre* "man") or *prissa* (= *prisa* "the hurry") do not appear in the Modern Spanish lexicon, and therefore they need to be marked as [noun]. The following SQL statement limits the query to just those forms that are not annotated – i.e. there is no value for the x1 (POS) field of the POS table (x_c)) (note that the references to the POS table are underlined). To actually update the POS table, it would simply be a matter of including an appropriate UPDATE command at the beginning of this SQL statement.

```
select top 200 x3.w2,sum(x3.x12)
        from x3, x_c where
        x3.w1 in ('un', 'una','vn', 'vna') and
        x3.w3 = 'que' and
        x3.w2 = x_c.w1 and
        x_c.x1 is null
        group by x3.w2
        having sum(x3.x12) > 2
        order by sum(x3.x12) desc
```

The preceding examples use the collocational information from the n-grams sequences to help assign part-of-speech. Another very useful feature of the relational database architecture is the ability to use sub-queries to perform multiple (essentially simultaneous) tests on the data to extract the relevant forms. For example, suppose that we want to identify the imperfect subjunctives of [–AR] verbs in Old Spanish, which are forms that end in [-ASSE]. The problem is that in addition to the imperfect subjunctives, there are many other word types that can end in [-ASSE], such as *uasse* "goes-away-3SG", *tornauasse* "came-back-3SG", and *nasse* "is-born". How can we separate out only the imperfect subjunctives? It turns out that the imperfect subjunctive has a regular morphological relationship to the infinitive. It is formed by adding [-ASSE] to the verbal root, which corresponds to the infinitive without the ending [-AR]. So to check whether an [-ASSE] form is really an imperfect subjunctive, we want to first check to see whether there is a related infinitive with fairly high frequency in Old Spanish. This can be done quite effectively with the following query:

```
select top 100 x12, w1
        from x1
        where w1 like '%asse' and len(w1) >=4 and
        left(w1,len(w1)-4) in
        (
        select left(w1,len(w1)-2)
        from x1 where
        w1 like '%ar' and len(w1) >=2
```

```
and x12 > 10
)
order by x12 desc
```

In addition to part-of-speech annotation, the n-grams tables can also be used to lemmatize word types that do not appear in the Modern Spanish lexicon. For example, the following query will list the 100 most frequent words (w1) in the 1200s (x12), where the word pattern is 'quer*' (*querya, querremos*), 'qu_er*' (*quiere, qujeren*) or 'quis*' (*quisyere, quise*) (Old Spanish forms of *querer* "to want"), which have a NULL lemma value (x1) in the lemma table [x_l]:

```
select top 100 x1.x12,x1.w1
    from x1,x_l
    where (x1.w1 like 'quer%' or x1.w1 like 'qu_er%' or x1.w1 like 'quis%') and
    x1.w1 = x_l.w1 and
    x_l.x1 is null
    order by x1.x12 desc
```

We may determine, however, that a number of the morphologically similar word forms actually belong to another lemma. In the example above, for example, a number of the matching forms belong to the Old Spanish verb *querellar* "to quarrel". To further limit the forms, we may wish to provide a syntactic context in which only the forms of *querer* "to want" will be found, such as those forms that precede an infinitive. The following query does this by using a sub-query to limit the [*que*-] forms to just those that occur immediately before an infinitive (w2; [-AR/-ER/-IR/-YR] for Old Spanish) in the table of 2-grams (x2), which is a syntactic environment in which *querellar* "to fight" would likely not occur:

```
select top 100 x12,w1 from x1
where (w1 like 'quer%' or w1 like 'qu_er%' or w1 like 'quis%')
and w1 in
(
select w1 from x2
where (w1 like 'quer%' or w1 like 'qu_er%' or w1 like 'quis%') and
(w2 like '%ar' or w2 like '%er' or w2 like '%ir' or w2 like '%yr')
)
order by x12 desc
```

In summary, by combining pattern matching, collocational information for multi-word n-grams tables, and frequency information for each of these n-grams, it is possible to annotate hundreds or thousands of words in just a few hours, even when there is no lexicon available.

8 Modular architecture and unlimited annotation

The final advantage of the relational database / n-grams approach is also perhaps the most important one. In this approach, the n-grams/frequency tables are simply one part of the overall database, albeit the most important part. But because they are in a relational database, they can very easily be joined to other tables within the same database, and there is no limit to the amount of annotation that can be applied to the corpus.

An example of one of these "auxiliary" databases is a table in the Corpus del Español that contains the synonyms for 30,000 words. Users can submit a simple query like either of the two that follow:

```
!destruir
    !destruir.*
```

A simple query like [!destruir] "to destroy" will retrieve a listing of all of the synonyms for the word (such as *romper, derrotar, deshacer, derribar*, and seventeen other verbs), along with the frequency of each of these words in each of the sub-corpora. In the second case, [!destruir.*], the frequency and distribution of all of the forms of all of the synonyms are retrieved, such as (*romper, rompe, rompieron, deshizo, deshaga*, etc). These can also be re-grouped by lemma, to show the overall frequency of all of the forms for each of the synonyms. In addition, after seeing a list of synonyms, users can click on any one of the words in the new list, and see a new list of synonyms for that word, and thus create "semantic chains" of related words. In essence, this is like a traditional electronic thesaurus, with the important difference that users can see the frequency and distribution of each synonym.

The important point is that this database (and any number of other databases) can easily be linked to the main n-grams / frequency database, to create even more powerful queries. An example of this is the following query:

[SEARCH] !decir.* !chiste.*
[SEARCH] !mandar.* que *.v_subj_ra

In terms of the underlying SQL command, the following refers to the first example [!decir.* !chiste.*]:

```
select * from x2 where
    w1 in (select w1 from x_L where x1 in ('decir', 'mencionar', 'aseverar', 'exponer' . . .)) and
    w2 in (select w1 from x_L where x1 in ('chiste', 'ocurrencia', 'chirigota', 'ingeniosidad'  . . . ))
    order by x19 desc
```

The first query searches for all forms of all synonyms of *decir* "to say" followed by all forms of any synonym of *chiste* "a joke", producing results like *conté chistes* "I-told jokes" or *diciéndose burlas* "telling-to-themselves jokes". The second query searches for all forms of all lemma that are synonyms of *mandar* "to order, command", followed by the subjunctive marker *que* "that", followed by a past subjunctive (*mandé que fueran* "I made them leave"; *hicieron que dijera* "they made her say").

Another example of linked tables and unlimited annotation in the Corpus del Español are the customized lists of words that users can create, which contain words that are related morphologically, syntactically, or semantically, such as adjectives describing emotions, words ending in [-azo] that denote a blow or strike, or a list of temporal adverbs. After they are created by the users, they are stored in tables where they can later be used as part of the query syntax. For example, suppose that a user [Jones] creates a list called [emotions], which contains a list of verbs of emotion (e.g. *gustar, alegrar, sorprender* "to please, make happy, surprise"). The user could simply retrieve this list and see the distribution and use of all of the words in the list. But they can also use this list as part of a more complex query, as in the following:

me/nos/te/os/le/les [Jones:emotions].* que *.v_subj_pres

This query searches for all cases of:

one of the indirect object pronouns [*me, nos, te, os, le, les*] + any form of any of the words in the customized [emotions] list created by [Jones] + *que* + present subjunctive

The web script then translates this into the following SQL command, which is passed to the database:

```
select top 300 * from x4 where
    w1 in ('me', 'nos', 'te', 'os', 'le', 'les') and
    w2 in (select w1 from x_l where x1 in ('gustar', 'sorprender', 'agradar', 'alegrar')) and
    w3 in ('que') and
    w4 in (select w1 from x_c where x1 in ('v_subj_pres'))
```

and will return a list like *me gusta que haya* "it please me that there is", *le sorprende que tengan* "it depresses him/her that they have".

The important point is that there can be an unlimited number of levels of annotation on the corpus – whether parts of speech, or lemma, or synonyms, or translations between languages, or etymologies, or customized lists, and these can all be linked together with simple SQL JOIN commands. It is not apparent how this degree of flexibility or power would be an inherent property of the standard scheme, in which the annotation is based within the textual corpus itself.

9 Conclusion

In summary, the use of large n-grams / frequency tables in relational databases provides corpus creators with a number of important advantages over traditional approaches:

1) It allows for extremely fast retrieval, because of the clustered indexes that can be created on the n-grams tables. For example, in the 100 million word Corpus del Español, most searches take less than one second, and even the most complex queries – involving morphological pattern matching, lemma, part-of-speech, and synonyms – usually take less than five seconds.
2) Because the annotation resides in relational database tables, it can quickly and efficiently be updated, without having to traverse the entire textual corpus and update tags within the text itself.
3) The n-grams / frequency tables allow users to access frequency information directly, such as rank-ordered lists of words with a particular morphological pattern, those synonyms that occur in one register by not in

another, or the most common strings for a particular syntactic construction, which occur with a set frequency in different sub-corpora.

4) The n-grams tables also allow for complex comparisons involving sub-queries, such as which nouns occur with only one of two verbs or adjectives, or which words are synonyms of two different verbs (such as *to jump* and *to run*) or two different nouns (such as *floor* and *level*).

5) The n-grams can also be a powerful tool to help annotate the corpus itself and to help build the lexicon, through the use of sub-queries and the collocational frequencies that can easily be extracted from the tables. This is particularly useful in the annotation of corpora that have minimal or non-existent lexicons.

6) Finally, the modular approach, which relies on linked relational databases, allows both the corpus creator and the end user to join together an unlimited number of annotation tables (or user-defined databases) of virtually any complexity, with no decrease in performance.

References

Berglund Y 1999 Exploiting a Large Spoken Corpus: An End-User's Way to the BNC. *International Journal of Corpus Linguistics* 4:29-52.

Biber D, Conrad S, Reppen R 1998 *Corpus linguistics: investigating language structure and use.* Cambridge, Cambridge University Press.

Clear J 1993 The British National Corpus. In Landow G, Delany G (eds), *The Digital Word: Text-Based Computing in the Humanities*, Cambridge, MIT Press, pp 163-87.

Clear J, Fox G, Francis G, Krishnamurthy R, Moon R 1996. COBUILD: The State of the Art. *International Journal of Corpus Linguistics* 1:303-14.

Kennedy G 1998 *An Introduction to Corpus Linguistics*. London, Longman.

McEnery A, Wilson A 1996 *Corpus Linguistics*. Edinburgh, Edinburgh University Press.

Olsen M, Hinkelman E 1991 *Problems of Large Literary Databases: Morphological Analysis of ARTFL* (CILS Technical Report). Chicago, University of Chicago.

Rocha P, Santos D 2000 CETEMPúblico: Um corpus de grandes dimensões de linguagem jornalística portuguesa. In *Actas do V Encontro para o processamento computacional da língua portuguesa escrita e falada (PROPOR'2000),* São Paulo, pp. 131-140.

Santos D, Bick E 2000 Providing Internet access to Portuguese corpora: the AC/DC project. In *Proceedings of the Second International Conference on Language Resources and Evaluation (LREC 2000),* Athens, pp 205-210.

Sinclair J 1987 *Looking Up. An Account of the Cobuild Project in Lexical Computing*. London, Collins ELT.