Interpreting Plurals in the Naproche CNL

Marcos Cramer and Bernhard Schröder

University of Bonn and University of Duisburg-Essen
cramer@math.uni-bonn.de, bernhard.schroeder@uni-due.de
http://www.naproche.net

1 Introduction

The Naproche CNL [2] is a controlled natural language for mathematical texts, i.e. a controlled subset of the semi-formal language of mathematics (SFLM) as used in mathematical journals and textbooks. The Naproche system translates Naproche CNL texts first into Proof Representation Structures (PRSs, [2]), an adapted version of Discourse Representation Structures, which are further translated into lists of first-order formulae which are used for checking the logical correctness of a Naproche text using automated theorem provers.

The two main applications that we have in mind for Naproche are to make formal mathematics more readable to the average mathematician, and to use it as a tool that can help undergraduate students to learn how to write formally correct proofs and thus get used to (a subset of) SFML.

A recent addition to the Naproche CNL is the possibility to make plural statements. By this we mean not only statements involving nouns in the plural (e.g. "numbers") and verbs conjugated in plural forms (e.g. "are"), but also conjunctions of noun phrases (e.g. "$x + y$ and $x \cdot y$ are even"). We discuss the collective-distributive ambiguity as well as a special scope ambiguity conjunctions give rise to, and explain how these ambiguities are resolved in Naproche. Plural definite noun phrases (e.g. "the real numbers") are not yet implemented in Naproche and are left out of the discussion in this abstract.

2 Collective vs. distributive readings of plurals

The following sentence is ambiguous:

(1) Three men lifted a piano.

It can mean either that three men lifted a piano together (in a single lifting act), or that there were three lifting acts, each of which involved a different man lifting a piano. The first is called the *collective* reading, the second the *distributive* reading. The ambiguity arises because the agent of a lifting event can either be a collection of individuals or a single individual.

In SFLM, both the collective and the distributive reading exist:

(2) 12 and 25 are coprime.

(3) 2 and 3 are prime numbers.

Instead of (2), one could also say "12 is coprime to 25." So the adjective "coprime" can be used in two grammatically distinct ways, but in both cases refers to the same mathematical binary relation: either it is (predicatively or attributively) attached to a plural NP that gets a collective reading, or it has as a complement a prepositional phrase with "to". When used in the first way, we call "coprime" a *collective adjective*, when used in the second way, a *transitive adjective*. We say that the two logical arguments of "coprime" are *grouped* into one collective linguistic argument, a plural NP with collective reading. In general, mathematical adjectives expressing a symmetric binary relation have these two uses (cf. "parallel", "equivalent", "distinct", "disjoint"; in the case of "distinct" and "disjoint", the preposition used for the transitive case is "from" rather than "to"). Other cases of grouped arguments are "x and y commute" (cf. "x commutes with y") and "x connects y and z" (cf. "x connects y to z"). "x is between y and z" is an example of an expression with a grouped argument for which there is no corresponding expression without grouped arguments.

Since "prime number" expresses a unary relation, it is not possible to group two of its logical arguments into a single linguistic argument; this explains why (3) can't have a collective reading of the sort that (2) has.

We know of no example where a mathematical expression has a linguistic argument that can be either a collectively interpreted plural NP or a singular NP (and can hence also be a distributively interpreted plural NP), and could therefore give rise to an ambiguity like that of (1).

3 Scope ambiguity

Another kind of ambiguity of special interest for our treatment of plurals and noun phrase conjunctions is a scope ambiguity that arises in certain sentences containing a noun phrase conjunction and a quantifier:

(4) A and B contain some prime.

(4) can mean either that A contains a prime and B contains a (possibly different) prime, or that there is a prime that is contained in both A and B. In the first case we say that the scope of the noun phrase conjunction "A and B" contains the quantifier "some", whereas in the second case we say that the scope of "some" contains the noun phrase conjunction. We call the first reading the *wide-conjunction-scope* reading and the second the *narrow-conjunction-scope* reading.

The conjunction scope can be disambiguated semantically as in (5) and (6).

(5) x and y are integers such that some odd prime number divides $x + y$.

(6) x and y are prime numbers p such that some odd prime number q divides $p + 1$.[1]

[1] Given that this example is made up, one might ask whether it really occurs in SFLM texts that a plural noun followed by a variable is predicatively linked to a conjunction

(5) only has a narrow-conjunction-scope reading, because the existentially introduced entity is linked via a predicate ("divides") to a term ("$x + y$") that refers to the conjuncted noun phrases individually. (6) on the other hand only has a wide-conjunction-scope reading.

In general, there is a strong tendency in SFLM texts to resolve scope ambiguities by giving wider scope to a quantifier that is introduced earlier in a sentence than to a quantifier introduced later in the sentence. This is a principle that we have already long ago adopted into Naproche in order to avoid scope ambiguities in the Naproche CNL. With the addition on NP conjunctions and disjunctions, we extended this principle to their scopes, with the exception of the cases like (5) where another reading is forced by certain syntactical considerations.

4 Pairwise interpretations of collective plurals

In SFLM texts, one often sees sentences like (7) and (8), which are interpreted in a pairwise way as in (9) and (10):

(7) 7, 12 and 25 are coprime.

(8) All lines in A are parallel.

(9) $\mathit{coprime}(7, 12) \wedge \mathit{coprime}(12, 25) \wedge \mathit{coprime}(7, 25)$

(10) $\forall x, y \in A\ (x \neq y \rightarrow \mathit{parallel}(x, y))$[2]

Sometimes, especially in connection with the negative collective adjectives "distinct" and "disjoint", this interpretation is reinforced through the use of the word "pairwise", in order to ensure that one applies the predicate to all pairs of objects collectively referred to by the plural NP. But given that this pairwise interpretation is at any rate the standard interpretation of such sentences even in the absence of the adverb "pairwise", we decided not to require the use of the word "pairwise" in the Naproche CNL.

The Naproche CNL allows only this pairwise interpretation for a plural NP that is used as a grouped argument of such a collective adjective.[3] (11) is a sentence where another reading might naturally be preferred. However, it seems to us that such sentences hardly appear in real mathematical texts.

(11) Some numbers in A and B are coprime.

of terms as in this example. One real example that we found comes from page 4 of [1]: "Notice that 13, 37, 61, . . . , are primes p such that p^3+2 and p^3+1 are squarefree."

[2] The distinctness condition here can be ignored in the case of symmetric relations like "parallel", but is certainly needed for non-reflexive relations like "coprime" or "disjoint".

[3] Which adjectives classify as *collective adjectives* is coded into the lexicon of the Naproche CNL.

5 The plural interpretation algorithm

The PRS construction algorithm for the representation of single sentences is similar to the standard threading algorithm for DRS construction (see [3]), and is implemented in Prolog. This algorithm has been adapted in order to cope with plurals, plural ambiguity resolution and pairwise interpretations as explained in the previous sections. We illustrate how the algorithm treats plurals by considering the following example sentence:

(12) x and y are distinct primes p such that $2p+1$ is a square number and some odd prime divides $x+y$.

The algorithm works by first producing a preliminary representation (left PRS in Fig. 1): Here the NP conjunction gets a plural discourse referent (p in Fig. 1), which is linked to the discourse referents of the conjuncts by a *plural_dref condition*. We give the NP conjunction wide scope over all quantifiers introduced later, and all assertions made in the scope of the plural NP are inserted in a special *plural sub-PRS*. Now a number of processes yield the final PRS:

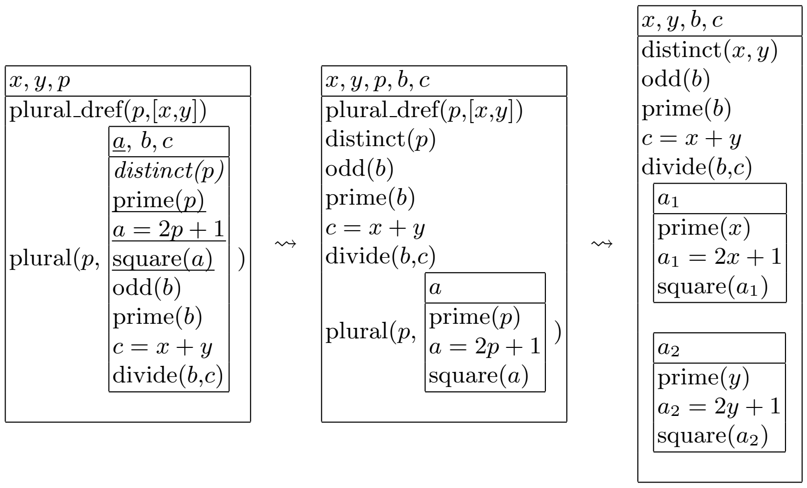

Fig. 1. Preliminary stages and final stage in the PRS construction for sentence (12).

1. In the plural sub-PRS, we mark every PRS conditions which consists of a predicate that has the plural discourse referent as grouped argument ("distinct(p)" in Fig. 1, marked by italicisation)
2. In the plural sub-PRS, we recursively mark (in Fig. 1 by underlining) all PRS conditions that weren't marked in step 1 and contain the plural discourse referent or a marked discourse referent, and all discourse referents contained in a PRS marked in this way, until no more conditions and discourse referents can be marked by this process.
3. All discourse referents and PRS conditions in the plural sub-PRS not marked in step 2 get pulled out of the plural sub-PRS and inserted into its super-PRS (yielding the second PRS in Fig. 1).

4. For every PRS condition $p(d)$ with grouped argument d, and every pair d_1, d_2 of distinct discourse referents linked to d via a plural_dref condition, we create a PRS condition of the form $p(d_1, d_2)$ and remove the original PRS condition $p(d)$ (in our example this amounts to replacing "distinct(p)" by "distinct(x, y)").
5. For every discourse referent d linked to the plural discourse referent p, we make a copy of the plural sub-PRS in which every instance of p is replaced by d, removing the original plural sub-PRS (right PRS in Fig. 1).

6 Related and Future Work

The syntax of Attempto Controlled English (ACE) allows plurals, which are interpreted in ACE in an unambiguous way. The disambiguation used by ACE is very distinct from Naproche's: while Naproche gives preference to distributive and wide-conjunction-scope readings, ACE allows only collective and narrow-conjunction-scope readings, unless the word "each" is used. This difference is due to the fact that for Naproche we focused on the interpretations common in SFLM, whereas ACE took the English language as a whole into account. Our focus on mathematical language also made it important for us to give "x and y are coprime" and "x is coprime to y" the same representation, which ACE does not do.

ForTheL, the controlled natural language of the System for Automated Deduction (SAD), a project with similar goals to Naproche, already included the two uses of words like "parallel" and "to commute" and produced the same representation no matter in which way they were used [5].

At the moment, Naproche does not yet allow anaphoric pronouns like "it" and "they". When Naproche is extended to allow them, some rules specifying how to control the many ways in which an anaphoric antecedent for "they" can be chosen (see [4]) will have to be specified and implemented, again with special attention to existing usage in SFLM.

References

1. Cohen, G. L.: Derived Sequences. Journal of integer sequences, Vol. 6 (2003)
2. Cramer, M., Fisseni, B., Koepke, P., Kühlwein, D., Schröder, B., Veldman, J.: The Naproche Project – Controlled Natural Language Proof Checking of Mathematical Texts. CNL 2009 Workshop, LNAI 5972 proceedings (2010, in press)
3. Johnson, M., Klein, E.: Discourse, anaphora and parsing, Proceedings of the 11th coference on Computational linguistics (1986)
4. Kamp, H., Reyle, U.: From Discourse to Logic: Introduction to Model-theoretic Semantics of Natural Language, Kluwer Academic Publisher (1993)
5. Paskevich, A.: The syntax and semantics of the ForTheL language (2007)
6. Schwertel, U.: Controlling Plural Ambiguities in Attempto Controlled English (ACE), Proceedings the 3rd International Workshop on Controlled Language Applications (2000)