Internal and External Evidence in the Identification and Semantic Categorization of Proper Names

David D. McDonald

mcdonald@cs.brandeis.edu

Abstract

We describe the proper name recognition and classification facility ("PNF") of the SPARSER natural language understanding system. PNF has been used very successfully in the analysis of unrestricted texts in several sublanguages taken from online news sources. It makes its categorizations on the basis of 'external' evidence from the context of the phrases adjacent to the name as well as 'internal' evidence within the sequence of words and characters. A semantic model of each name and its components is maintained and used for subsequent reference.

We describe PNF's operations of delimiting, classifying, and semantically recording the structure of a name; we situate PNF with respect to the related parsing mechanisms within Sparser; and finally we work through an extended example that is typical of the sorts of text we have applied PNF to.

1 Introduction: Internal versus External Evidence

It has long been appreciated by people working with proper names in unrestricted written texts that any adequate treatment requires the use of a grammar. We know at this point that the correct identification and semantic categorization of names requires an analysis based on the patterning of orthographic and lexical classes of elements that is analogous to what one finds in the other content-rich, syntactic structure-poor phrase types in the 'periphery' of the language such as numbers, dates, citations, etc. The point of this paper will be to argue that this grammar must be context sensitive, and that it should incorporate a rich semantic model of names and their relationships to individuals.

The context-sensitivity requirement derives from the fact that the classification of a proper name involves two complementary kinds of evidence, which we will term 'internal' and 'external'. *Internal evidence* is derived from within the sequence of words that comprise the name. This can be definitive criteria, such as the presence of known 'incorporation terms' ("Ltd.", "G.m.b.H.") that indicate companies; or heuristic criteria such as abbreviations or known first names often indicating people. Name-internal evidence is the only criteria considered in virtually all of the name recognition systems that are reported as part of state of the art information extraction systems (see e.g. Rau 1991, Alshawi 1992, DARPA 1992), most of

which depend on large (~20,000 word) gazetteers and lists of known names for their relatively high performance.

By contrast, *external evidence* is the classificatory criteria provided by the context in which a name appears. The basis for this evidence is the obvious observation that names are just ways to refer to individuals of specific types (people, churches, rock groups, etc.), and that these types have characteristic properties and participate in characteristic events. The presence of these properties or events in the immediate context of a proper name can be used to provide confirming or criterial evidence for a name's category. External evidence is analyzed in PNF in terms of substitution contexts and operationalized in terms of context-sensitive rewrite rules.

External evidence is a necessity for high accuracy performance. One obvious reason is that predefined word lists can never be complete. Another is that in many instances, especially those involving subsequent references, external evidence will override internal evidence. In the final consideration it is always the way a phrase is used—the attributions and predications it is part of—that make it a proper name of a given sort; without the consideration of external evidence this definitive criteria is missed, resulting in mistakes and confusion in the state of the parser. (Relying solely on name lists has led to some funny errors, for example mistaking the food company *Sara Lee* for a person. Even some external evidence such as a title can be inadequate, if considered apart from the wider context of use, as in *General Mills*—both actual mistakes made by an otherwise quite reasonable program some years ago (Masand & Duffey 1985).)

An additional reason, and one with considerable engineering utility from the point of view of the grammar writer, is that the inclusion of external evidence into the mix of available analysis tools reduces the demands on the judgements one requires of internal evidence. It can provide a weaker (less specific) categorization about which it can be more certain, which can then be refined as external evidence becomes available. Lacking definitive internal evidence one can initially label a segment simply as a 'name', and then later strengthen the judgement when, e.g., the segment is found to be adjacent to an age phrase or a title and context-sensitive rewrite rules are triggered to re-label it as a person and to initiate the appropriate semantic processes.

This kind of staged analysis is a requirement when the conclusions from internal evidence are ambiguous. It is not uncommon, for example, to have the names of a person and a company in the same news article both share a word, i.e. when the company is named after its founder. A subsequent reference using just that word cannot be definitively categorized on internal evidence alone, and must wait for the application of external evidence from the context. (In the event that the context is inadequate, as when it involves a predication not in the grammar, such 'name' segments can be left to default judgements by statistical heuristics operating after a first pass by the parser, and the stronger categorizations then tested for coherency as the parse is resumed. In the case of a company and a person with the same name, a well edited publication is unlikely to use the ambiguous word to refer to the founder without prefixing it with "Mr." or "Ms." as needs be, so a word with both person and company denotations but without external evidence can be assumed with some assurance to be referring to the company.)

In this paper we will describe in some detail the proper name facility of the SPARSER natural language understanding system: "PNF", with particular attention to how it uses external evidence and deploys its semantic model of names and their referents to handle ambiguities such as the one noted just above.

In a blind test of an earlier implementation of PNF in the context of "Who's News" articles from the the Wall Street Journal, it performed at nearing 100% in the scored sentences in the

sublanguage for which a full grammar had been prepared. We are currently testing a new implementation on a more diverse set of texts.

Space will not permit a comparison of this algorithm with other approaches to proper names beyond occasional remarks and references. As far as we know this is the only treatment of proper names that makes essential use of context-sensitive rewrite rules, however the FUNES system of Sam Coates-Stephens (1992) is very similar to this work in making essential use of external evidence, and Coates-Stephens's extensive research into proper names is an important contribution to the field; we have adopted some of his terminology as noted below.

2 An overview of the procedure: Delimit, Classify, Record

The goal of the proper name facility in Sparser (PNF) is to form and interpret full phrasal constituents—noun phrases—that fit into the rest of a text's parse and contribute to the analysis of the entire text just like any other kind of constituent. That is, PNF is operating as a component in a larger natural language comprehension system, and not as a standalone facility intended for name spotting, indexing, or other tasks based on skimming. This integration is essential to the way the PNF makes its decisions; it would not operate with anything like the same level of performance if it were independent, since there would then be no source of external evidence.

To form full constituents for use in a language comprehension system we must (1) *delimit* the sequence of words that make up each name, i.e. identify its boundaries; (2) *classify* or categorize the resulting constituent based on the kind of individual it names; and (3) *record* the name and the individual it denotes in the discourse model as our interpretation of the constituent's meaning.

For other parts of Sparser's grammar, these three actions are done with one integrated mechanism much as they would be in any other system. Constituents are initiated bottom up by the terminal rules of Sparser's lexicalized grammar, and then compositions of adjacent constituents are checked for and nonterminal nodes introduced in accordance with Sparser's moderately complex control structure that permits a deterministic parse and monotonic semantic interpretation. The rules these operations deploy (essentially standard productions, though their mode of operation is more like that of a categorial grammar) both delimit and classify (label) constituents in one action: the categories are given by the production's lefthand sides, and the new constituents' boundaries by the sequence of (typically binary) daughter constituents on the rules' righthand sides, with the new constituent's denotation given by an interpretation function included directly with the rule and applied as the rule completes.

This normal mode of operations has not proved workable for proper names, and the reason has to do with the central problem with names from the point of view of a grammar, namely that in unrestricted texts the set of words that names can be comprised of cannot be completely known in advance. The set is unbounded, growing at an apparently constant rate with the size of one's corpus, while the growth of other classes of content words tapers off asymptotically (Liberman 1989). This means that we cannot have a lexicalized grammar for proper names since the bulk of the names we will encounter will be based on words that are undefined at the time the grammar is written.

Complicating the picture is the fact that virtually any normal word can do double duty as part of a name (" ... *Her name was equally preposterous. April Wednesday, she called herself,*

and her press card bore this out. " MacLean 1976 pg.68). This means that one either introduces a massive and arbitrary ambiguity into one's normal vocabulary, allowing any word to be part of a name, or one looks for another means of parsing proper names, which is the course that was taken with SPARSER's PNF, where we separate out the three actions into distinct and largely independent operations. In the remainder of this section we will sketch the procedures for delimiting, classifying, and recording proper names; then in the following section we will go into detail with an example once the parsing machinery that the procedures draw on has been introduced.

Looking first at the kind of formal mechanisms used, the delimit operation is based on a simple state machine, rather than the application of context free rewrite rules as done in the rest of the grammar. This reflects that fact that the internal constituent structure of a proper name is far more flat than hierarchical, and consequently should be treated as a Kleene Star structure in a regular grammar (<name-word>+). The binary branching tree that one would get with context-free rules would be an artifact of the rule application machinery rather than reflect the grammar of names.

In essence, the delimitation algorithm simply treats as a group any contiguous sequence of capitalized words (including 'sequences' of length one). This is virtually always the correct thing to do as the example below illustrates, though the exceptions have to be treated carefully as discussed later.

> *"The Del Fuegos, O Positive, and We Saw the Wolf will perform acoustic sets in Amnesty International USA Group 133's Seventh Annual Benefit Concert at 8 p.m. on Friday, March 19, at the First Parish Unitarian Universalist Church in Arlington Center."* (Arlington Advocate, 3/18/93)

A sequence is terminated at the first non-capitalized word[1] or comma; other punctuation is handled case by case, e.g. "&" is taken to extend sequences and periods are terminators unless they are part of an abbreviation.

Classifying a proper name is a two-step process. First, Sparser's regular parsing routines are applied within the delimited word sequence. This introduces any information the grammar has about words or phrases it knows. This information supplies the basis for the bulk of the structure within a proper name, and provides the name-internal evidence on which the classification will be based. For Sparser it includes:

- embedded references to cities or countries, e.g. "*Cambridge Savings Bank* ".
- open class 'keywords' like "*Church*" or "*Bank*" (following Coates-Stephens terminology), and the incorporation-terms used by companies of various countries when giving their full legal names ("*Inc.*" in the U.S.A., "*P.T.*" in Indonesia, "*G.m.b.H.*" in Germany, etc.).

---

[1] It is reasonable to depend upon the existence of mixed-case text, since the number of online sources that supply uppercase only is rapidly diminishing and will probably disappear once all of the Model-33 Teletypes and other 6-bit data entry terminals in the world are finally junked. In any event, to handle all-uppercase texts within Sparser's design it is only the delimitation algorithm that must be changed, and a good approximation of the needed segmentation is independently available from the distribution of function words and punctuation in any text. In the example above these are the commas, the appostrophe-s, "*in*", "*at*", and "*on*"; a mistake would be made in "*We Saw the Wolf*" [sic] which will be problematic without external context in any event.

- the relatively closed class of stylized modifiers used with people like "*Jr.*", "*Sr.*", "*Mr.*", "*Dr.*".
- items used for heuristic classification judgements (the items above are definitive) such as abbreviations (a strong indicator that the name refers to a person or a company based on a person's name), or punctuation like "*&*" or ambiguous modifiers like "*II*" (which invariably means 'the second', but may be used with Limited Partnerships as well as people).

The parsing stage will reveal when the capitalized word -based delimitation has exceeded its proper scope. One such case is of course when a proper name appears just after the capitalized word at the start of a sentence: "*An Abitibi spokesman said ...*". This is handled by defining all the closed-class grammatical functional words as such so that they will be seen during this embedded parse, and resegmenting the word sequence to exclude them.

Another, more interesting case is where we have a sequence of modifiers prefixed to a proper name that are themselves proper names, e.g. "*Giant Group said ... seeking to block a group led by Giant Chairman Burt Sugarman from acquiring ...*". In this situation there is no hope for correctly separating the names unless the grammar includes rules for such companies and titles, in which case they will appear to the classifier as successive edges with the appropriate labels so that it can know to appreciate them for what they are and to leave them out. (It is perhaps a matter of judgement to hold that a person's title is not a part of their name, but that policy appears to be the most consistent overall since it permits the capitalized premodifier version of a title of employment (e.g. "*Chairman*") and its predicative lowercase version (as in an appositive) to be understood as the same kind of relationship semantically—a different one than the relationship between a person and their conventional title such as "*Mr.*".) It is important to appreciate that all of these considerations only make sense when one is analyzing proper names in the context of a larger system that already has grammars and semantic models for titles and employment status and such; and they are hard to justify in an application that is simply name spotting.

In practice, the operations of delimiting and classifying are often interleaved, since the classification of an initially delimited segment can aid in the determination of whether the segment needs to be extended, as when distinguishing between a list of names and a compound name incorporating commas, e.g. "*... a string of companies – including Bosch, Continental and Varta – have announced co-operative agreements ...*" (The Financial Times, 5/16/90); v.s. "*HEALTH-CARE FIRM FOLDS: Wood, Lucksinger & Epstein is dissolving its practice.*" (Wall Street Journal 2/26/91). We will describe this process in the extended example at the end of the paper.

Once the words of the sequence have been parsed and edges introduced into the chart reflecting the grammar's analysis, the second part of the classification process is initiated as a state machine is passed over that region of the chart to arrive at the most certain classification possible given just this name-internal evidence. If no specific conclusion can be reached, then the sequence will be covered with an edge that is simply given the category 'name', and it will be up to external evidence to improve on that judgement as will be described later. If a conclusion is made as to the kind of entity being named then the edge will be labeled with the appropriate semantic category such as 'person', 'company', 'newspaper', etc.

The *recording* process now takes over to provide a denotation for the edge in the discourse model. Before this denotation is established, the representation of the name is just a label and a

designated sequence of words and edges internal to the name (e.g., edges over an embedded reference to a city or region). What we are providing now is a structured representation of the name qua name—a unique instance of one of the defined classes of names that reifies that specific pattern of words and embedded references.

Including names as actual entities in the semantic model, rather than just treating them as ephemeral pointers to the individuals they name and only using them momentarily during the interpretation process, provides us with an elegant treatment of the ambiguity that is intrinsic to names as representational entities. Real names, unlike the hypothetical 'rigid designators' entertained by philosophers, may refer to any number of individuals according to the contingent facts of the actual world. We capture this by making the denotation of the lexico-syntactic name—the edge in the chart—be a semantic individual of type 'name' rather than (the representation of) a concrete individual. The name object in turn may be then associated in the discourse model with any number of particular individuals of various types: people, companies, places, etc. according to the facts in the world. Thus the ambiguity of names is taken not to be a linguistic fact but a pragmatic fact involving different individuals having the same name.

The structure that the semantic model imposes on names is designed to facilitate understanding subsequent references to the individuals that the names name. The type of name structure used predicts the kinds of reduced forms of the name that one can expect to be used. This design criteria was adopted because, again, the overarching purpose of PTF is to contribute to the thorough understanding of extended unrestricted texts, and this means that it is not enough just to notice that a given name has occurred somewhere in an article (which is easy to do by just attending to the cases where the full company name is given with the 'incorporation term' that well edited newspapers will always provide when a company is introduced into a text, e.g. "*Sumitomo Electric Industries, Ltd.*"). Instead, PTF must be able to recognize that the same individual is being talked about later when it sees, e.g., "*Sumitomo Electric*" (or "*the company*"), as well as to distinguish it from subsequent references to other companies that share part of the name: "*Sumitomo Wiring Systems*"; or to correctly deduce a subsidiary relationship "*Sumitomo Electric International (Singapore)*". Similarly, people and companies or locations that share name elements should be appreciated as such: "*the Suzuki Motors Company ... Osamu Suzuki, the president of the company*".

To facilitate subsequent reference, not only does each proper name receive a denotation as an entirety, but the words that comprise it are also given denotations which are related, semantically, to the roles the words have each played in that name and in the names of other particular individuals. Thus the word "*Suzuki*", for example, is taken to always denote the same semantic object, prosaically printed as #<name-word "suzuki">. In turn this individual is related to (at least) two other individuals—to the car company by way of the relation 'first-word-in-name', and to its president by the relation 'family-name'.

3 The setting for the process

In order to supply the external evidence needed to accurately categorize proper names and understand them semantically, a language understanding system must include grammars (and their attendant semantic models) for properties and event-types that are characteristically associated with them, and they should have as broad a coverage as possible.

SPARSER has been applied to understanding news articles about people changing their jobs (particularly the Wall Street Journal's "Who's News" column), and with a lesser competence to articles on corporate joint ventures and quarterly earnings. As a result, it has quite strong semantic grammars for some of the very most frequent properties of companies and people in business news texts: the parent–subsidiary relationship between companies, age, titles, and for a few of the more common event-types (via its primary grammars).

A complementary consideration is such relatively mundane things as what approach will be taken to punctuation, capitalization, or abbreviations. For SPARSER, since it is designed to work with well-edited news text written by professional journalists, punctuation is retained and there are grammar rules that appreciate the (sometimes heuristic) information that it can provide. The whitespace between words is also noted (newlines, tabs, different numbers of spaces) since it provides relatively reliable evidence for paragraphs, tables, header fields, etc., which in turn can provide useful external evidence.

Additionally, SPARSER is designed to handle a constant, unrestricted stream of text, day after day, and this has led to a way to treat unknown words that allows it to look at their properties, and from that possibly form them into proper names, without being required to give them a long-term representation which would eventually cause the program to run out of memory.

To illustrate how these aspects work, and at the same time establish the implementation setting in which proper name processing takes place, we will now describe the lower levels of SPARSER's operation, starting with its tokenizer and populating the terminal positions of the chart.

3.1 Tokenizing

The tokenizer transduces characters to objects representing words, punctuation, digit sequences, or numbers of spaces. It is conservatively designed, just grouping contiguous sequences of alphabetic characters or digits and punctuation and passing them them all through to be the terminals of the chart, where even the simplest compounds are assembled by sets of rules that are easily changed and experimented with. For example, rather than conclude inside the tokenizer that the character sequence "$47.2 million" is an instance of money, it just passes through six tokens, including the space.

A word is 'known' if it is mentioned in any of the rules of the grammar.[2] A known word has a permanent representation, and the tokenizer finds and returns this object when it delimits the designated sequence of characters. The 'token-hood' of this word type is represented by the word object filling particular places in the chart.

---

[2] Note that since this is a lexicalized semantic grammar, words have preterminal categories like 'title' or 'head-of-CO-phrase' (e.g. "*company*", "*firm*", "*enterprise*") or are often treated just as literals, e.g. all the prepositions or words like "*based*".

The tokenizer separates the identity of a word from its capitalization. A word is defined by the identity of its characters. The pattern of upper and lowercase letters that happens to occur in a given instance is a separate matter, and is represented in the chart rather than with the word.[3] Thus when the chart is populated with terminals, each position records the word that starts there, its capitalization, and the kind of whitespace that preceded it, all given as separate fields of the position object. The scan is done incrementally in step with the rest of the SPARSER's actions.

3.2 Word-triggered operations

Sparser's processing is organized into layers. Tokenizing and populating the terminals of the chart is the first level, then comes a set of special operations that are triggered by words or their properties (e.g. ending in "*ed*" or consisting solely of digits). The application of phrase structure rules is the next layer, and finally there is the application of heuristics to try spanning gaps caused by unknown words. Semantic interpretation is done continuously as part of what happens when a rule completes and an edge is formed. We will not describe the last two layers (see McDonald 1992 for a description of the phrase structure algorithm), but will briefly describe the word-level operations since it includes triggering PNF.

Operations triggered by the identify of a word include forming initials and known abbreviations, and particularly the recognition of multi-word fixed phrases which we call "polywords" following Becker (1975). Polywords are immutable sequences of words that are not subject to syntactically imposed morphological changes (plurals, tense) and that can only be defined as a group. Polywords are a natural way of predefining entities that have fixed, multi-word names such as the countries of the world, the states of the US, major cities, etc. Instances of this relatively closed class of individuals are a valuable kind of evidence in the classification of proper names.

When PNF finishes the recognition and classification of a new name, it adds to the grammar a polyword rule for the sequence of words in the name, with the recorded name-object as the polyword's denotation, so that the process can be short-circuited the next time the name is seen. Note that this does not stop PNF from triggering and running its delimiting operation the next time that sequence is seen; it only speeds up the classification and recording. If we allowed the polyword operation to take precedence, we would never see the longer word sequences that embed known names ("*New York Port Authority*"), or we would have to resort to a more complex algorithm.

We have a particularly fast algorithm for checking for polywords when the first word of the sequence has been seen. There are also special rules that allow paired punctuation to be grouped (parentheses, quotations, etc.) even if the words separating them are not all known. This is particularly useful for picking up nicknames embedded within a person's name since it will often be an unknown word in quotations inside parentheses ("*Richard M. ("tricky Dick") Nixon*"). Subsidiaries of companies are often marked for their geographical area in the same way, e.g. "*manufactured by UNIVERSAL FLUID HEADS (Aust.) PTY. LTD.*" (from the name plate on a camera tripod).

---

[3] One can deliberately define a capitalization-sensitive version of a word, e.g. to syntactically distinguish titles in pre-head position from those in appositives or elsewhere. In such cases there is a distinct word object with a link to the case-neutral version of the word.

The first check at the word-level is for actions triggered by a word's properties, particularly here the properties of its characters. This is how compound numbers are formed (42,357.25) triggering off words that are sequences of digits, and it is how PNF is triggered. Every time a chart position is reached that indicates that the following word is capitalized, PNF is called. PNF then takes over the process of scanning the successive terminals of the chart, until it scans a word that is not capitalized, deliberately calling other SPARSER mechanisms like polyword recognition or phrase structure rewrite rules as needed.

When PNF is finished, its results are given in an edge it constructs over the sequence of capitalized words and selected punctuation, with the label on the edge dictating how it will fit into SPARSER's later processing layers and the referent field of the edge pointing to the name object that records it in the discourse history. Since Sparser uses a semantic grammar, the label is the constituent's classification—a semantic category like 'person'. There is also conventional label (always NP for a name) included with the edge for default or heuristic rules of phrasal formation; see McDonald (submitted) for the details of this two label system.

4 Walking through an example

In this final section of this paper we will look at the processing of the following paragraph-initial noun phrase from the Wall Street Journal of 10/27/89, article #34:

> *"An industry analyst, Robert B. Morris III in Goldman, Sachs & Co.'s San Francisco office, said ... "*

The capitalization of the very first word "An" triggers PNF, whose delimitation process stops immediately with the next word since it is lowercase. The classification pass through the (one word) sequence then shows it to be a grammatical function word, and classification applies the rule 'single word sequences consisting solely of a non-preposition function word are not to be treated as names'. PNF is then finished; the article reading of "*An*" will have been introduced into the chart during classification; and the scan moves on.[4]

As the parse moves forward, the title phrase "an industrial analyst" is recognized and the comma after it is marked as possibly indicating an appositive (or also a list of titles, though this is less likely).

PNF is triggered again by the capitalization of "*Robert*", and the delimitation process takes it up to the word "*in*". Running the regular rules of the grammar within that sequence uncovers the abbreviation and the generation-indicator "III" for 'the third'. We do not maintain any lists of the common first names of people or such, so consequently both "*Robert*" and "*Morris*" are

---

[4] Fortunately we have yet to see a company whose name was "*The*"—one wonders how journalists would deal with it. Of course there are companies like Next Inc. and On Technology, which, like the names of race horses or boats, add spice to the grammarian's life by overloading the interpretations of closed-class words. The only consistent treatment we have arrived at for these ("*On*" referring to the company does occur in sentence-initial position) is to treat the words as ambiguous and to introduce two edges into the chart, one for each reading. We only do this if the full name of the company appeared earlier in the article, however, when the preposition will have received its denotation as an element of a name and the basis of the ambiguity been established.

seen as unknown words. The abbreviation and generation-indicator are enough, however, to allow the sequence to be reliably classified as the name of a person.

Given that classification, an edge is constructed over the sequence and given the label 'person', and the recording process constructs a name object for the edge's denotation. The pattern given by the classifier is 'name – initial – name – generation-indicator', which is clear enough for the name subtype 'person's name with generation' to be instantiated. This object takes a sequence of first names or initials, a last name (the word at the end before "III"), and then the "III" in a slot that also gets words like "*Junior*". Let us call this new name-individual Name-1.[5]

Part of the recording is the creation of denotations for the words "*Robert*" and "*Morris*". Individuals are created for them of type 'single word element of a name', and rules are added to the grammar so that the next time we see them in the grammar we will be taken directly to those same individuals. By warrant of forming the interpretation of the whole sequence as Name-1, we can now attribute properties to the names (semantic objects) 'Robert' and 'Morris'— 'Robert' is the first name of Name-1 and 'Morris' is the last name. The policy of letting words like "*Morris*" denote name objects with semantic links to the name they are part of (with that name in turn linked to the person whose name it is) provides a very direct way to understand subsequent references involving just part of the original name (e.g. "*Mr. Morris*") as we can trace it directly to the person just by following those links. (Of course the links will also take us to anyone else the world model knows of who has that same last name, hence the need for a good discourse model that appreciates the context set up by the article being processed.)

Moving on, the next point where PNF comes into play is at the word "*Goldman*". It is seen as a one word sequence because of the comma just after it, and is an unknown word. Not being a function word, it is spanned with an edge labeled just 'name' and recorded with just a new single-word name individual as its denotation. Given the significance of commas for name patterns, we also at this point make a note (set a variable) that this comma is preceded by a name.

PNF immediately resumes with "*Sachs & Co.*", stopping the delimitation process when it recognizes the "'" and "*s*" tokens as constituting an appostrophe-s, which is a definitive marker for the end of a noun phrase. During the delimitation process, the abbreviation "*Co.*" will also have been recognized and expanded, and the "*&*" noted and appreciated as being a punctuation mark that can appear in names. Punctuation is always handled during the course of delimitation for just these reasons.

The presence of the "*&*" and the word "*Company*" are definitive markers of companies and the classification process will start the assembly of a pattern to send off to be recorded. In this case however, as noted earlier, there is what amounts to an interaction between classification and delimitation. Part of what the classifier knows about companies is the profusion of cases where the name of the company is a sequence of words separated by commas (law firms, advertising agencies, any sort of partnership tends to use this name pattern). Appreciating this, the process looks for the contextual note about the preceding comma. Finding it, it observes that the name in front of the comma is not itself classified as a company (which would have indicated a list of companies rather than a single name), and it proceeds to assimilate the edge over "*Goldman*" and the comma into the name it is already assembling. Had there been still

---

5 There is no interesting limit on the number of 'first names' a person can have, so we have not yet found it profitable to have any more structure in that field than simply an ordered sequence; consider "*M.A.K. Halliday*", "*(Prince) Charles Philip Arthur George*".

more 'stranded' elements of the name followed by commas, this would have been noted as well and those elements added in.

Occasionally the name of a company like this is given with "*and*" instead of the special punctuation character. Had that happened here, the fact that the "*and*" preceded the word "*company*" would have been sufficient evidence to take the whole sequence as the name of a single company, however if there had been no such internal evidence within any of the elements of the conjunction, they would have been grouped together as unconnected names spanned by a single edge labeled 'name', leaving it to external evidence from the context to supply a stronger categorization (both as category and whether they were one name or several), as we can see with the next capitalized word sequence that PNF delimits, "*San Francisco*".

With access to a good gazetteer we could have already defined San Francisco as the name of a city using a polyword. Alternatively, without needing any word list we can conclude that it is a location, and probably a city, just by looking at its context: the word "*office*".

As said earlier, the availability of mechanisms that use external evidence like this allows PNF to make a weak analysis that can be strengthened later. In this case it will see "*San Francisco*" as a sequence of two unknown words. Without any internal evidence to base its judgement on, it can only (1) accept the sequence as a phrase and span it with an edge, indicating that the words have more relationship to each other than either individually has to its neighbors, and (2) give this edge the neutral label 'name'.

After PNF is done, the phrase structure component of Sparser takes over. Sparser's rewrite rule facility includes context-sensitive as well as context-free productions, including for this case the rule

```
name -> location /  ___ "office"
```

That is, an edge labeled 'name' can be respanned with a new edge with the label 'location' when the name edge appears just in front of the word "office". Context sensitive rules are handled with the same machinery as context free rules, e.g. the trigger pattern here is the same as that of a CF rule with the righthand side 'name' + "*office*". The difference is simply that instead of covering the whole righthand side with a new edge we just respan the one indicated constituent.

Similarly, if the person in this example had the name "*Robert Morris*" (rather than "*Robert B. Morris III*"), where there would have been no available internal evidence to indicate its classification during the operations of PNF, we would later have applied either of two context free rules: one working forwards from the definitively recognized title, the other backwards from the pp 'in-company'.

```
name -> person / title "," ____
name -> person / ____ in-company
```

A repertoire of such context-sensitive rules or their equivalent is needed if a proper name classification facility is expected to work well with the open-ended set of name words found in actual texts; Sparser used a set of roughly 30 rules to handle the names in the blind test on the Who's News column mentioned earlier.

5 References

Alshawi, Hiyan (ed.) (1992) *The Core Language Engine*, MIT Press.

Becker, Joe (1975) "The Phrasal Lexicon", in Schank & Webber (eds.) Proc. TINLAP-1, ACM, 60-63.

Coates-Stephens, Sam (1992) "The Analysis and Acquisition of Proper Names for the Understanding of Free Text", *Computers in the Humanities.*

(DARPA) Defence Advanced Research Projects Agency (1992) *Proceedings of the Fourth Message Understanding Conference: MUC-4* , June 1992, Morgan Kaufmann.

Liberman, Mark (1989) Panel presentation at the 27th Annual Meeting of the ACL, Vancouver.

MacLean, Alistair (1976) *The Golden Gate*, Fawcett Publications, Greenwich Connecticut.

Masand, Brij M. & Roger D. Duffey (1985) "A Rapid Prototype of an Information Extractor and its Application to Database Table Generation", working paper, Brattle Research Corporation, Cambridge, MA.

McDonald, David D. (1992) "An Efficient Chart-based Algorithm for Partial-Parsing of Unrestricted Texts", proceedings of the 3d Conference on Applied Natural Language Processing (ACL), Trento, Italy, April 1992, pp. 193-200.

_____ (submitted) "The Interplay of Syntactic and Semantic Node Labels in Partial Parsing", manuscript submitted to the International Workshop on Parsing Technologies, August 1993.

Rau, Lisa F. (1991) "Extracting Company Names from Text", proc. Seventh Conference on Artificial Intelligence Applications, February 24-28, 1992, IEEE, 189-194.