Quantitative Narrative Analysis Software Options Compared: PC-ACE and CAQDAS (ATLAS.ti, MAXqda, and NVivo)

Roberto Franzosi, Emory University, Professor.
Sophie Doyle, Oxford University, Dr.
Laura McClelland, Emory University, Dr.
Caddie Putnam Rankin, Emory University, Dr.
Stefania Vicari, University of Leicester, Dr.

Corresponding author: Stefania Vicari
email: sv32@le.ac.uk,
phone number: +44 (0)7580155396
fax number: +44(0)116 252 5276

Abstract

This paper shows how to carry out Quantitative Narrative Analysis (QNA) with different text analysis software (PC-ACE, Program for Computer-Assisted Coding of Events, and various CAQDAS programs, Computer-Assisted Qualitative Data Analysis Software: ATLAS.ti, MAXQDA, and NVivo). QNA is a methodological approach to narrative texts that exploits invariant properties of narrative (namely, a "story grammar", based on actors, actions, and their attributes) to make a statistical analysis of words possible. In comparing PC-ACE and CAQDAS, the paper leads the reader through the steps involved in setting up a grammar, in data entry, and in data query. A careful comparison of limits and possibilities of the two types of software will allow the reader interested in QNA to make an informed choice between a full implementation of QNA in a specialized but unknown software (PC-ACE) and a limited implementation in any of the widely used and popular CAQDAS programs.

Key Words

CAQDAS; PC-ACE; Quantitative Narrative Analysis; software; story grammar.

Quantitative Narrative Analysis (QNA)

Quantitative Narrative Analysis (QNA) is a social science research method for the analysis of narrative texts (Franzosi, 2010). Like content analysis, the method traditionally used in the social sciences to extract quantitative information from texts, QNA is quantitative. It aims at turning words into numbers (Franzosi, 2004). It does so via a coding scheme (a "story grammar"), made up of coding categories (the objects of the grammar), to which coders assign specific parts of a text. By computing word frequencies of specific coding categories, words are then turned into numbers – numbers to be analyzed with the help of statistical tools. In content analysis, the coding categories reflect the substantive/theoretical interests of the investigator (Franzosi, 2008). As a result, these categories change from study to study, albeit with some accumulation of categories within specific areas of investigation (e.g., content analysis of health issues rather than content analysis of media advertisements or of news). Contrary to content analysis, however, QNA's coding categories and coding schemes are based on invariant, structural properties of narrative, namely the sequential organization of narrative (in story and plot) and its simple surface linguistic structure of subject-verb-object (where subjects are typically social actors and verbs actions).

Story Grammars

Linguistically, a story is typically expressed as the simple micro-level structure Subject-Verb-Object (SVO). The SVO structure is a general form of the language, and not just of narrative (indeed, "the canonical form of the language"); but in narrative, both Subject and Object are typically social actors or organizations (the Object can also be a "thing", i.e., a physical object) and the Verb is a verb of doing or saying (i.e., a social action). In narrative, the SVO structure has also been referred to as a "story grammar."[1]

A story grammar can be as simple as the basic three elements of the "semantic triplet" SVO – the subject, the action, and the object of the action – or very complex, with the addition of a number of modifiers for each element of the SVO (such as type, number, organization, name and last name of the Subject and Object and time, space, reason, outcome, instrument of the Verb). Thus, a story grammar broadly corresponds to the 5 Ws of journalism – Who, What, When, Where, Why – with the potential addition of several more elements (on the 5 Ws, see Franzosi, in press).

Relationships between the various objects (or coding categories) of a story grammar can be expressed formally with the help of "rewrite rules." Through a rewrite rule, we can express the simple SVO structure (or semantic triplet) in terms of its basic components:

<semantic triplet> → {<subject>} {<verb>} [{<object>}]

where the symbol → refers to a rewrite rule (or production), whereby an element to the left of the rule can be rewritten in terms of the elements to its right.[2] Each element of the triplet can then be further rewritten, down to its "terminal" symbols (those found in the language itself. In this instance, a lynching in Hinesville, Georgia):[3]

<subject> → {<actor>} {<characteristics>}

<actor> → crowd | mob | posse | Negro | Sheriff |...

<characteristics> → <type> <number> <organization> <name> …

…

<verb> → <verbal phrase> <circumstances>

<verbal phrase> → bring | burn | shoot | kill | hang |...

< circumstances > → <time> <space> <reason> <instrument> outcome>

…

<object> → {<actor>} | {<physical object>}

The user might also need to set up *alternative objects* (e.g., both <semantic triplet> and <alternative semantic triplet>), particularly when the narrative information is extracted from different documents. This allows one to code the different ways of telling the same story, depending upon the story teller's point of view.

Franzosi (see for all, 2004: 59-61, 2010: 31-32) has shown how a grammar of this kind can provide coding schemes with more desirable properties than traditional content analysis coding schemes: 1. they are based on invariant structural properties of narrative rather than on the investigator's ad-hoc substantive or theoretical interests; 2. they allow for the setting up of both hierarchical and relational links between the objects of the grammar (or coding categories of the coding scheme, in the language of content analysis); 3. they provide coded output that preserves much of the input narrative information; 4. the narrative flavor of coded output delivers more reliable data, since coded output must make sense to any "competent user of the language" (semantic coherence).

QNA with Lynching Stories

To understand how a story grammar can help structure narrative information, consider the following newspaper article published in the *Atlanta Constitution* on February 9, 1888.

[EXCERPT 1 HERE]

The article of Excerpt 1 is a good example of narrative text, made up of clauses characterized by factual events, with a series of actors engaged in different actions. Overall, the text provides minimal description and evaluation.[4] The article gives us the events ("fire", "arrest", "lynching"), the events' time ("a few weeks ago, "yesterday", "last night") and place ("Hinesville", "jail", "woods"), the actors involved ("Negro", "deputy Sheriff", "band of armed

men"), the actor's actions ("destroyed", "burned", "overpowered, "shot", "arrested", "charge", "committed to jail", "confessed", "implicated", "carrying"), the targets of their actions, as actors ("Negro") or physical objects ("house" and "warehouse").

In a second article published the next day (February 10, 1888), also by the *Atlanta Constitution,* we are given additional and different details on the same events:

[EXCERPT 2 HERE]

If we combine the factual information from the two articles, we obtain a full narration of the event story (information from the later article in italics):

[EXCERPT 3 HERE]

This "combined"[5] lynching story would look as follows within the coding categories of a complex story grammar (in black, the name of the coding categories, in blue the information taken from the newspaper articles, in green, the alternative coding of a semantic triplet; semantic triplets have been ordered chronologically; Participant-S, Process, Participant-O reflect Halliday's language for SVO (Halliday, 1994):

(Macroevent) (Victim: unnamed Negro)

(Event 1: (Type of event: burglary)

(Semantic Triplet 1.1: (Participant-S: (Actor: (Individual: (Name of individual actor: Negro) (Personal characteristics: (Gender: male) (Race: Negro)))) (Actor: (Collective actor: (Name of collective actor: individuals) (Collective characteristics: (Number: (Quantitative value: (Numeric value: 4))))))) (Process: (Simple process: (Verbal phrase: broke into) (Aggregate action: violence against things) (Circumstances: (Time: (Date: (Indefinite date: (Temporal direction: before) (Reference yardstick: **(Event 2)**)))) (Space: (Territory: (Type of territory: (County: Liberty))))))) (Participant-O: (Physical object: (Type of physical object: store) (Ownership: (Individual: (Name of individual actor: individual) (Personal characteristics: (First name and last name: (Last name: Chapman)) (Gender: male)))))))

(Semantic Triplet 1.2: (Participant-S: (Actor: (Individual: (Name of individual actor: Negro) (Personal characteristics: (Gender: male) (Race: Negro)))) (Actor: (Collective actor: (Name of collective actor: individuals) (Collective characteristics: (Number: (Quantitative value: (Numeric value: 4))))))) (Process: (Simple process: (Verbal phrase: stole) (Aggregate action: illegality) (Circumstances: (Time: (Date: (Indefinite date: (Temporal direction: before) (Reference yardstick: **(Event 2)**)))) (Space: (Territory: (Type of territory: (County: Liberty))))))) (Participant-O: (Physical object: (Type of physical object: everything) (Ownership: (Individual: (Name of individual actor: individual) (Personal characteristics: (First name and last name: (Last name: Chapman)) (Gender: male)))))))

(Event 2: (Type of event: fire)

(Semantic Triplet 2.1: (Participant-S: (Actor: (Individual: (Name of individual actor: incendiary)))) (Process: (Simple process: (Verbal phrase: destroyed) (Aggregate action: violence against things) (Circumstances: (Time: (Date: (Indefinite date: (Time expression: (Quantitative qualifier: a few) (Generic temporal expression: weeks) (Temporal direction: ago)) (Reference yardstick: (Article date: 02/08/1888))))) (Space: (City: (City name: Hinesville) (County: Liberty) (State: Georgia))) (Instrument: (Type of instrument: kerosene))))) (Participant-O: (Physical object: (Type of physical object: house))) (Participant-O: (Physical object: (Type of physical object: warehouse)) (Alternative triplet: **(Semantic Triplet)**))))

(Alternative Semantic Triplet) (Participant-S: (Actor: (Individual: (Name of individual actor: Negro) (Personal characteristics: (Gender: male) (Race: Negro)))) (Actor: (Collective actor: (Name of collective actor: individuals) (Collective characteristics: (Number: (Quantitative value: (Numeric value: 4))))))) (Process: (Simple process: (Verbal phrase: set fire) (Aggregate action: violence against things) (Circumstances: (Time: (Date: (Indefinite date: (Time expression: (Quantitative qualifier: a few) (Generic temporal expression: weeks) (Temporal direction: ago)) (Reference yardstick: (Article date: 02/09/1888))))) (Space: (City: (City name: Hinesville) (County: Liberty) (State: Georgia))) (Instrument: (Type of instrument: kerosene))))) (Participant-O: (Physical object: (Type of physical object: store) (Ownership: (Individual: (Name of individual actor: individual) (Personal characteristics: (First name and last name: (Last name: Chapman)) (Gender: male)))))) (Participant-O: (Physical object: (Type of physical object: warehouse) (Ownership: (Organization: (Name of organization: Savannah Florida and Western Railway))) (Participant-O: (Physical object: (Type of physical object: buildings) (Number: (Qualitative value: (Numeral: several))))))

(Event 3: (Type of event: Investigation)

(Semantic Triplet 3.1: (Participant-S: (Actor: (Individual: (Name of individual actor: Sheriff)))) (Process: (Simple process: (Verbal phrase: investigated) (Aggregate action: investigation) (Circumstances: (Action type (Adverb): preliminarly) (Time: (Date: (Indefinite date: (Time expression: (Generic temporal expression: yesterday)) (Reference yardstick: (Article date: 02/08/1888))))) (Space: (City: (City name: Hinesville) (County: Liberty) (State: Georgia))) (Reason: (Name of reason: fire) **(Semantic Triplet 2.1)**))))))

(Semantic Triplet 3.2: (Participant-S: (Actor: (Individual: (Name of individual actor: Sheriff)))) (Process: (Simple process: (Verbal phrase: charged) (Aggregate action: accusation) (Circumstances: (Time: (Date: (Indefinite date: (Time expression: (Day: Tuesday)) (Reference yardstick: (Article date: 02/09/1888))))) (Space: (City: (City name: Hinesville) (County: Liberty) (State: Georgia))) (Reason: (Name of reason: fire) **(Semantic Triplet 2.1)**))))) (Participant-O: (Actor: (Individual: (Name of individual actor: Negro) (Personal characteristics: (Gender: male) (Race: Negro))))))

(Semantic Triplet 3.3: (Participant-S: (Actor: (Individual: (Name of individual actor: Sheriff)))) (Process: (Simple process: (Verbal phrase: arrested) (Aggregate action: control) (Circumstances: (Time: (Date: (Indefinite date: (Time expression: (Day: Tuesday)) (Reference yardstick: (Article date: 02/09/1888))))) (Space: (City: (City name: Hinesville) (County: Liberty) (State: Georgia))) (Reason: (Name of reason: charge) **(Semantic Triplet 3.2)**))))) (Participant-O: (Actor: (Individual: (Name of individual actor: Negro) (Personal characteristics: (Gender: male) (Race: Negro))))))

(Semantic Triplet 3.4: (Participant-S: (Actor: (Individual: (Name of individual actor: Magistrate)))) (Process: (Simple process: (Verbal phrase: committed to jail) (Aggregate action: control) (Circumstances: (Time: (Date: (Indefinite date: (Time expression: (Generic temporal expression: yesterday)) (Reference yardstick: (Article date: 02/08/1888))))) (Space: (City: (City name: Hinesville) (County: Liberty) (State: Georgia)))))) (Participant-O: (Actor: (Individual: (Name of individual actor: Negro) (Personal characteristics: (Gender: male) (Race: Negro))))))

(Semantic Triplet 3.5: (Participant-S: (Actor: (Individual: (Name of individual actor: Negro) (Personal characteristics: (Gender: male) (Race: Negro))))) (Process: (Simple process: (Verbal phrase: confessed) (Aggregate action: communication) (Circumstances: (Time: (Date: (Indefinite date: (Time expression: (Generic temporal expression: yesterday)) (Reference yardstick: (Article date: 02/08/1888))))) (Space: (City: (City name: Hinesville) (County: Liberty) (State: Georgia))) (Content: (Name of content: deed) **(Semantic Triplet 2.1)**))))))

(Semantic Triplet 3.6: (Participant-S: (Actor: (Individual: (Name of individual actor: Negro) (Personal characteristics: (Gender: male) (Race: Negro)))))) (Process: (Simple process: (Verbal phrase: implicated) (Aggregate action: accusation) (Circumstances: (Time: (Date: (Indefinite date: (Time expression: (Generic temporal expression: yesterday)) (Reference yardstick: (Article date: 02/08/1888))))) (Space: (City: (City name: Hinesville) (County: Liberty) (State: Georgia)))))) (Participant-O: (Actor: (Collective actor: (Name of collective actor: individuals) (Collective characteristics: (Number: (Qualitative value: (Numeral: several)))))))))

(Event 4: (Type of event: lynching)

(Semantic Triplet 4.1: (Participant-S: (Actor: (Individual: (Name of individual actor: deputy Sheriff) (Personal characteristics: (Gender: male))))) (Process: (Simple process: (Verbal phrase: was going) (Aggregate action: movement/search) (Circumstances: (Time: (Date: (Indefinite date: (Time expression: (Day: yesterday)) (Reference yardstick: (Article date: 02/08/1888)))) (Time of day: (Indefinite time of day: (Moment of the day: night)))) (Space: (City: (Spatial direction: to) (Locality within city: (Building: (Proper name of building: Hinesville jail) (Type of building: jail))) (City name: Hinesville) (County: Liberty) (State: Georgia)))))) (Relation to next triplet: when))

(Semantic Triplet 4.2: (Participant-S: (Actor: (Collective actor: (Name of collective actor: band) (Collective characteristics: (Gender: male) (Type of actor (Adjective): armed) (Number: (Quantitative value: (Numeric value: 15))))))) (Process: (Simple process: (Verbal phrase: overpowered) (Aggregate action: violence against people) (Circumstances: (Time: (Date: (Indefinite date: (Time expression: (Generic temporal expression: yesterday)) (Reference yardstick: (Article date: 02/08/1888)))) (Time of day: (Indefinite time of day: (Moment of the day: night) (Space: (City: (City name: Hinesville) (County: Liberty) (State: Georgia))))))) (Participant-O: (Actor: (Individual: (Name of individual actor: deputy Sheriff) (Personal characteristics: (Gender: male))))))

(Semantic Triplet 4.3: (Participant-S: (Actor: (Collective actor: (Name of collective actor: band) (Collective characteristics: (Gender: male) (Type of actor (Adjective): armed) (Number: (Quantitative value: (Numeric value: 15))))))) (Process: (Simple process: (Verbal phrase: carried) (Aggregate action: violence against people) (Aggregate action: search/movement) (Circumstances: (Time: (Date: (Indefinite date: (Time expression: (Generic temporal expression: yesterday)))) (Reference yardstick: (Article date: 02/08/1888)))) (Time of day: (Indefinite time of day: (Moment of the day: night)) (Space: (City: (Spatial direction: to) (Locality near city: woods) (City name: Hinesville) (County: Liberty) (State: Georgia)))))))

(Semantic Triplet 4.4: (Participant-S: (Actor: (Collective actor: (Name of collective actor: band) (Collective characteristics: (Gender: male) (Type of actor (Adjective): armed) (Number: (Quantitative value: (Numeric value: 15))))))) (Process: (Simple process: (Verbal phrase: killed) (Aggregate action: violence against people) (Circumstances: (Time: (Date: (Indefinite date: (Time expression: (Generic temporal expression: yesterday))))) (Reference yardstick: (Article date: 02/08/1888)))) (Time of day: (Indefinite time of day: (Moment of the day: night)) (Space: (City: (Locality near city: woods) (City name: Hinesville) (County: Liberty) (State: Georgia))) (Instrument: shots)))) (Alternative triplet: **(Semantic Triplet)**))))

(Alternative Semantic Triplet: (Semantic Triplet:
(Participant-S: (Actor: (Collective actor: (Name of collective actor: band) (Collective characteristics: (Gender: male) (Type of actor (Adjective): armed) (Number: (Quantitative value: (Numeric value: 15))))))) (Process: (Simple process: (Verbal phrase: hung) (Aggregate action: violence against people) (Circumstances: (Time: (Date: (Indefinite date: (Time expression: (Day: Tuesday)) (Reference yardstick: (Article date: 02/09/1888)))) (Time of day: (Indefinite time of day: (Moment of the day: night)))) (Space: (City: (City name: Hinesville) (County: Liberty) (State: Georgia) (Alternative triplet: **(Semantic Triplet)**))))))))

(Alternative Semantic Triplet: (Semantic Triplet:
(Participant-S: (Actor: (Collective actor: (Name of collective actor: band) (Collective characteristics: (Gender: male) (Type of actor (Adjective): armed) (Number: (Quantitative value: (Numeric value: 15))))))) (Process: (Simple process: (Verbal phrase: burned) (Aggregate action: violence against people) (Circumstances: (Time: (Date: (Indefinite date: (Time expression: (Day: Tuesday)) (Reference yardstick: (Article date: 02/09/1888))))

(Time of day: (Indefinite time of day: (Moment of the day: night)))) (Space: (City: (Locality near city: woods) (City name: Hinesville) (County: Liberty) (State: Georgia)))))) (Participant-O: (Actor: (Individual: (Name of individual actor: Negro) (Personal characteristics: (Gender: male) (Race: Negro))))))

As the coded output shows, all narrative information found in the original input text also appears in the output, assigned to appropriate categories in the grammar.[6]

QNA's Questions

Coded output also shows another thing: the centrality of social actors and social action. QNA is all about actors and actions. After all, in narrative, the S and V components of the SVO structure are social actors and social actions (verbs of doing and saying). Such actor-centered methodological approach leads to research questions about social reality that are fundamentally different from the more traditional variable-centered approaches (see Abell, 2004; Franzosi, 2004: 240-42). Consider the literature on lynching. Variable-oriented explanations have focused on the role of such variables as the size of black population in a county, black crime rate, deflated price of cotton on the number of lynching events (for an excellent example, see Beck and Tolnay, 1990). QNA questions focus on the characters of a story (see Franzosi, 2010: 74). Who are the various characters in a story (the Negro, the Sheriff, the mob, the outraged women)? Which traits do they possess (male, female, young, old, good, evil)? What role do they play in the story? What do they do? Do any/some of the characters benefit (or suffer) from the actions of the other? When did the actions narrated in the story happen? Where? Were certain types of action committed by specific actors, at specific times or locations?

A Thousand Lynching Stories: "No software, no QNA"

The lynching reported by the *Atlanta Constitution* on February 9 and 10, 1888, (Excerpts 1 and 2) is one of nearly 400 lynchings that occurred in Georgia between 1875 and 1930. 1332 articles from 212 different newspapers reported these cases of lynching.[7] How, then, can one carry out

QNA on such large volume of documents? In the absence of fully automated approaches to the parsing of narrative, the only answer is: with computer-assisted QNA. The complexity and detail of the coding scheme and the nature of the "data" (basically, words rather than numbers – what we are typically accustomed to think as data) may otherwise relegate the use of story grammars as coding schemes to small-scale examples. The application of QNA for large socio-historical research requires the implementation of QNA in a computer environment. Indeed, "No software, no QNA", as Franzosi writes (2010: 67).

In the next sections, we will discuss two different software approaches to QNA: PC-ACE (Program for Computer-Assisted Coding of Events) and CAQDAS programs (Computer-Assisted Qualitative Data Analysis Software), in particular, ATLAS.ti, NVivo, and MAXQDA.[8] [9] PC-ACE (Program for Computer-Assisted Coding of Events, available for free download at www.pc-ace.com), is a software program specifically designed to carry out large-scale quantitative narrative analysis. It is based on a Relational Database Management System (RDMS) design, with information stored in separate tables, each table containing the values of a specific coding category (or object of the grammar, e.g., triplets, actors). Information stored in separate tables is linked via "relations" (i.e., overlapping fields) and queried using Structured Query Language (SQL). SQL relies on a handful of commands: *select* (i.e., extract a specific field), *from* (a specific table), *where* (filter on specific values). The SQL *count* command is used to compute the number of occurrences of a specific value in an object (coding category), either taken by itself or in relation to other objects (e.g., actors and their actions) (Franzosi 2010: 82).

Contrary to PC-ACE, CAQDAS programs were not originally designed for QNA, but as tools to aid in the analysis of qualitative research questions that explore theoretical concepts and provide support for theoretical development. Unlike PC-ACE, CAQDAS programs are tools for

qualitative analysis. CAQDAS programs do, however, have extensive query capacities that help uncover complex patterns or relationships within the data. CAQDAS query tools can even provide frequency counts, but the query tools are designed to highlight thematic and conceptual patterns across a number of different documents (e.g., in-depth interviews, focus groups, life histories).

It may seem invidious to propose a comparison between PC-ACE and CAQDAS programs, given the different research questions and design strategies of the two types of software. It is not. The point is to show whether (and how) it is possible to carry out QNA in CAQDAS programs. After all, given the popularity of these software programs, they are likely to be an investigator's first port of call.

Below, for both PC-ACE and CAQDAS programs, we focus on the three main tasks of QNA: 1. setting up a story grammar (using PC-ACE grammar objects/coding categories or CAQDAS codes); 2. data entry; and 3. data querying.

Quantitative Narrative Analysis in PC-ACE (Program for Computer-Assisted Coding of Events)

Setting up the Grammar

In order to perform QNA using PC-ACE, the first step is to setup the story grammar to be used as a coding scheme. In PC-ACE, users can set up complex story grammars made up of both relational objects (the Subject, the Verb, the Object, and each of their modifiers) and hierarchical objects (those that assemble the basic SVO narrative units into larger units, i.e., the Macroevent, the event, and the semantic triplet itself). Objects can further be simplex or complex, where simplex objects are rewritten as terminal symbols (i.e., as the words found in the dictionary of the language) and complex objects in terms of other simplex and/or complex objects. Hence, in

the grammar presented above, <actor> and <verbal phrase> are simplex objects; <subject>, <characteristics>, <verb>, <circumstances>, and <object> are complex objects (for the complete grammar used in the Georgia lynching project, see Appendix II).

While all the objects of a story grammar are relational (i.e., they are all related to one another via a set of rewrite rules), PC-ACE complex objects can also be set up as hierarchical nodes of the grammar. By using hierarchical objects, the actions of a story (the basic semantic triplets) can be aggregated into larger units (<events>); and events can be further aggregated hierarchically into even larger units (<Macroevent>). To set up a PC-ACE story grammar the user will need to generate all its complex and simplex objects, from the top hierarchical object (e.g., Macroevent) down to the lower objects (e.g., any of the actor characteristics or circumstances of verbs). PC-ACE provides a series of "setup forms" for the user to enter all the relevant information.

[FIGURE 1 HERE]

Figure 1 shows an example of PC-ACE setup form for the object "semantic triplet".[10] The right-hand sub-form titled "Reference/alias Children of Underlying Complex …" lists all the "children"[11] objects of a semantic triplet, with complex objects identified by a +, and optional objects by square brackets [ ] and multiple objects by curly brackets {}. From the setup form in Figure 1, the user could edit or delete any of the displayed objects, move up to their parent and down to their children, add new objects, simplex or complex, and view all instances of coded data for a selected object. When all the elements of the story grammar have been setup, data entry can start.

PC-ACE Data Entry

The text documents to be analyzed in PC-ACE are external to the program. They may be in

printed form, microfilms, or computer files (in which case links to these files can be stored within PC-ACE and used to open and analyze the documents from within PC-ACE). Even when the documents are in the form of computer files imported in PC-ACE, the data cannot be coded directly from the document itself, selecting portions of the text and assigning them to specific objects (coding categories). Rather, relevant information is entered manually into text boxes or combo boxes (see Figure 2), and document links only provide a convenient way to open and view a specific document.[12]

[FIGURE 2 HERE]

Each time a new piece of textual data is entered, PC-ACE adds it to its internal dictionary.[13] For stories narrated in multiple documents (e.g., the Hinesville lynching story told by two different newspaper articles), information can be taken and coded from any document that makes up the story in PC-ACE. This means that PC-ACE does not require any prior rewriting of multiple-document stories into a single document. New information is coded as encountered in each new document and automatically cross-referenced to that document.

PC-ACE Data Query

With the information coded into the appropriate objects of the grammar of data collection (coding categories in the content analysis terminology) and stored in PC-ACE's underlying relational database tables, we can run queries to investigate the lynching narrative, asking questions about that story. To make the process of data retrieval (or querying) easier, PC-ACE comes with a Query Manager based on a GUI (Graphical User Interface).[14] The GUI translates a query based on the objects of a user-defined story grammar and graphically displayed on the screen into an SQL query based on the PC-ACE tables and fields where data are stored[15]. For instance, to answer the general question "what did a certain actor do?" (e.g., "the Negro"), we

need to relate actors and actions under the same semantic triplet stored in the database and filter the actors on the value(s) we are interested in (i.e., Negro) (see Figure 3; fields highlighted in green will be shown in output; fields highlighted in red and green contain filters, i.e., the entries in an SQL WHERE clause).

[FIGURE 3 HERE]

The query yields the following list of available records showing the Negro's actions in the lynching narrative of Excerpt 3 (Figure 4).

[FIGURE 4 HERE]

This example shows that, with PC-ACE, we can easily answer the question: "WHO does WHAT?" (e.g., the mob). Similarly, we could answer questions such as: "WHEN and WHERE did these actions occur?" "WHO was the target of an agent's actions?"

Working with Aggregate Codes: PC-ACE

We used PC-ACE to carry out QNA on the 1,300 documents of the Georgia lynching project (1875-1930). The project yielded over 6,000 semantic triplets. Most dictionary values for the various simplex objects appear in this database with a frequency of 1, for a total of more than 200 distinct values for actors and 1,000 for actions when we look at the entire lynching database (a normal finding in this type of research; Franzosi, 2004: 83, 356). In order to facilitate statistical analysis, these distinct values should be aggregated into broader categories (Franzosi 2010: 103-4). Table 1 provides an example of aggregation applied to the actions found in the database.

[TABLE 1 HERE]

In the table, such codes as "violence against people" contain the individual action values "kill," "torture," "wound," "beat up," "assault," "rape," etc. Similarly, "movement" comprises values like "flee," "run away," "follow," "arrive," "plunge," etc.

However, how do we aggregate the original coding output into broader semantic clusters? PC-ACE offers two solutions to the problem of data aggregation: one during coding, with the coders carrying out data aggregation, and one after coding has been completed. In the first case (aggregation during data entry), the grammar would be setup to include a required simplex for the aggregate code of an object (e.g., aggregate actor or aggregate action). This would force a PC-ACE user to code both disaggregated and aggregated values for each dictionary entry during data entry. In addition, PC-ACE has an Update Manager that can be used after the completion of data collection to build new aggregate codes from a list of individual values via SQL UPDATE statements. With the help of the Update Manager, the researcher can set filters on any original code (see Figure 5) and define the new value to be indicated in the correspondent aggregate category (see Figure 6). Hence, the name of collective actors "daughters", "girls", "mothers", and "sisters" in Figure 5 can be reaggregated as "women" in a new aggregate category called aggregate actor, as shown in Figure 6.[16]

[FIGURE 5 HERE]

[FIGURE 6 HERE]

Hence, the PC-ACE user can conduct queries using both the original codes (e.g. name of collective actor) and their values (e.g., mothers, sisters, fathers, husbands) and aggregate codes (e.g., aggregate actor) and their values (e.g. women, men).

Quantitative Narrative Analysis in CAQDAS Applications

Having shown how to carry out QNA in PC-ACE, we now turn to illustrate how to perform these tasks in CAQDAS programs. Given the differences in software design, the order of implementation of the tasks changes slightly in CAQDAS compared to PC-ACE (grammar setup, data entry, and data query in PC-ACE and source document compilation, setup of codes, data coding, and data query in CAQDAS). This overview will not only show how to implement QNA using CAQDAS programs, but will also reveal the challenges (and ultimately, limitations) of doing QNA in CAQDAS. Because CAQDAS programs work in similar ways, we will use ATLAS.ti as a baseline, and discuss two other popular programs, MAXQDA and NVivo, only when significant differences apply.

Cross References: Grammar Objects and Documents

The first step towards carrying out data coding in CAQDAS programs is to import and link together the source documents[17] [18]. In ATLAS.ti, this is done through a "hermeneutic unit" (HU). A HU is used to link source documents together as one database so that the same code list is applied to all imported texts. In our case, the user would assign the two articles of our example to the same "hermeneutic unit" (HU)[19] (see Figure 7).

[FIGURE 7 HERE]

CAQDAS users can only code information one document at a time[20]. Each article within the HU is a separate document. Contrary to PC-ACE, this feature makes it difficult to link textual elements across different articles. Consider our two-article story of the Hinesville lynching brought together into a single HU. In the first article, we read that "a band of armed men overpowered the deputy Sheriff." In the second article we are told that the band ("a crowd") was composed of fifteen men ("a crowd of fifteen men"). In CAQDAS programs, there is no way of

linking this additional information to the actor coded under the first article. As a consequence, the fact that fifteen men (rather than a band with no numbers, as told in the first article) were involved would be lost.

To avoid losing this information, users would need to combine their source documents into a single document prior to coding. ATLAS.ti, NVivo, and MAXQDA allow the user to reword, manipulate, or modify source documents from within the programs. Alternatively, text revisions can be made outside the programs and the modified documents can then be imported into the database. Our lynching story, for instance, can be rewritten, combining the two available source documents of Excerpt 1 and 2 into a single story (Excerpt 3) (Appendix I). Needless to say, depending upon the number of articles in a project (1,300 in the Georgia lynching project) and the number of multiple-article events, this process of text editing could be very time consuming.

Setting up the Grammar in CAQDAS (Codes)

CAQDAS programs use codes to assign data to categories of analysis. Typically, these codes consist of key themes, concepts, processes or contexts based on theories from the researcher's academic tradition and the purpose of the analysis (Lewins and Silver 2007: 7). To conduct QNA in CAQDAS programs, the researcher needs to create codes that reproduce the objects of a story grammar, rather than key themes: the Participant-S, Process, and Participant-O (i.e., the SVO template) as well as all their modifiers (e.g. space, time).

[FIGURE 8 HERE]

Figure 8 shows the list of codes setup in ATLAS.ti for the lynching story. Codes are arranged alphabetically[21] as a seriatim list with no relational ties among them. As the entries in Figure 8 show, each code name includes: 1. the names of every parent object in the "path" of the

grammar all the way up to, but excluding, the first hierarchical object "semantic triplet" (e.g., the city within space, within circumstances, within simple process within process, in the code "Process: Simple process: Circumstances: Space: City"); 2. and, in addition, the actual value a code assumes (e.g., "Hinesville" for a city). Notice that since values have to be included in code names, unlike PC-ACE, codes cannot be generated prior to coding. This means that CAQDAS users will build their grammar while coding.

These code-naming criteria, requiring both paths and values, can lead to a very long code list. Consider the Georgia lynching database built in PC-ACE. The grammar consists of 79 simplex objects (i.e., objects that contain actual information taken from the source documents) and 67 complex objects (i.e., container objects that mark the path of a simplex object in the grammar) (see Appendix II). In CAQDAS programs the 79 simplex objects of the grammar can be easily reproduced as 79 codes. However, each CAQDAS code corresponding to a simplex object will also contain the name of that simplex's parent complex objects (e.g., the simplex object Name of individual actor can be reproduced in CAQDAS as "Participant-S: Actor: Individual: Name of individual actor"). The problem is that in the grammar of Appendix II several objects – both simplex and complex – appear as children of different parent complex objects. Thus, an Individual actor appears as the child of both Participant-S and Participant-O. The City appears as the residence of an Individual actor or a Collective actor (and each for either Participant-S or Participant-O), and as the location where an action occurs (one of the circumstances of the process). In the Georgia lynching grammar of Appendix II, 21 complex objects appear as children of more than one complex object.[22] Each of these 21 complex objects will require as many code names as the number of pathways they belong to.[23] In addition, in CAQDAS programs, the simplex children at the end of the path of each of these 21 complex

objects would also need to include the name of the different values the code for the simplex takes.[24] For instance, in the lynching database, there are 1266 distinct instances of Verbal phrase, yielding 1266 distinct codes (e.g., "Process: Simple process: Verbal phrase: arrest", "Process: Simple process: Verbal phrase: broke into", "Process: Simple process: Verbal phrase: burn"). Hence, in CAQDAS programs the code list will number in the thousands.

Code families provide a way to reduce the number of codes in ATLAS.ti.[25] [26] Code families are umbrella categories that group together individual codes – those that encompass the actual value a simplex takes. Thus, the codes "Process: Simple process: Verbal phrase: arrest" and "Process: Simple process: Verbal phrase: burn" can be grouped together into the code family "Process: Simple process: Verbal phrase". Figure 9 shows an example of ATLAS.ti's "code families".

[FIGURE 9 HERE]

Once a code family is created, the applicable codes need to be "assigned", i.e., they have to be attributed to the code family (see Figure 10).

[FIGURE 10 HERE]

Since the name of a code family does not appear in the code list (as one can see from Figure 8), the use of code families reduces the number of codes. In our case, with 79 simplex objects in the grammar, each resulting in a code with embedded path, if code families are applied for each simplex, this would result in a reduction of 898 codes (the number of simplex objects with their path but without the hard-coded value) – but still leaving perhaps thousands of codes embedding path and hard-coded values in the code list (1266 distinct values just for the Verbal phrase in our Georgia lynching database).

Working with Aggregate Codes in CAQDAS

To avoid working with such an unwieldy number of codes, even when using code families, in a CAQDAS approach to QNA, the only reasonable solution is to work with aggregate codes (and perhaps even with a simplified grammar). Such verbs as "kill", "wound", "burn", "riddle with bullets", "torture", "hang", "beat up", "lynch", "rape", … could all be coded as "violence". This way, the 1266 distinctive verbs can be reduced to a more manageable set of some 50 or 60 aggregated categories. The same is true for any other code (e.g., actors such as "police", "Sheriff", "deputy Sheriff", "officer", "marshal" etc. could all be coded as "law enforcement").

There are drawbacks to coding using aggregate codes. First, when coders perform both coding and aggregating at once, they dangerously come to play "surrogate scientist" (particularly if the aggregate codes are abstract theoretical categories, where not all values can be clearly pre-specified; see Markoff et al., 1975: 37). Second, some of the detailed original information is lost, and it can no longer be retrieved through a query (e.g. the frequency of verbs such as "kill" vs. "wound" once coded as "violence"). Third, the researcher would need to have the aggregate codes in mind at the start of the coding process. This, in turn, may require extensive pre-reading of the documents to be coded or a strong theory that provides the basis for aggregation regardless of dictionary values. Furthermore, findings may emerge during analysis that require unforeseen data queries. When these queries involve codes not incorporated into the original coding scheme, a CAQDAS researcher must choose between either creating new codes and re-reading and re-coding all documents or not performing the additional queries.[27] [28]

Setting up Codes for Hierarchical Objects (Macroevent and Event)

In narrative, the micro-level relational structure (the semantic triplet, or basic SVO, with S related to V and V related to O) can be aggregated into Macroevent-level hierarchical structures

(e.g., several triplets into an event, several events into a Macroevent). A story grammar, as we have seen, allows a user to specify both types of structures: relational and hierarchical. We have shown how CAQDAS users can set up a story grammar from the semantic triplet down. But such hierarchical objects as event and Macroevent in the lynching grammar have simplex children objects of their own (namely, <victim> and <type of event>).

In ATLAS.ti, one can set up these objects via the Primary Document Family Manager in the same way as for code families[29]. This tool allows one to generate a document family (i.e., a batch of documents with no internal hierarchical organization) and "assign" the relevant documents to the family. In so doing, texts can be grouped together by type of information (e.g., type of event at the event level).[30]

[FIGURE 11 HERE]

Essentially, a CAQDAS project is arranged into a hierarchy, as Figure 11 shows.[31] The highest level in the CAQDAS hierarchy consists of the documents. Beneath these come all the other branches of the hierarchy. The primary document branch is used for coding the simplex objects at the highest hierarchical levels of the story grammar (i.e., <victim> and <type of event>). The code branch is used for coding the objects found at the lower levels of the story grammar (i.e., <Participant-S>, <Process>, <Participant-O>, and their children).

There are benefits to applying different objects of the story grammar to different sections of the CAQDAS hierarchy, namely to document families and codes. First, by coding the highest hierarchical levels of the story grammar at the document level, in CAQDAS the codes for <type of event> and <victim> do not appear in the code list. This reduces the list of available codes – albeit, only marginally; but this depends upon the number of modifiers attached to the highest level of aggregation, in our case the "Macroevent". Second, and most importantly, it reduces the

time required for coding. Once the document families have been created it is a simple matter to move documents into a category. Entire documents, and consequently all the codes applied within those documents, are placed into a category depending upon the information they contain (the types of events they describe and the victim in those events). The alternative would be to apply another set of codes to every coded segment of text within the documents. This would take time and add to the complexity of coding.

Unlike ATLAS.ti, MAXQDA and NVivo allow the coding system to be displayed using a hierarchical view. Figure 12 shows that MAXQDA and NVivo display codes in the same way as PC-ACE, with indented objects indicating a child of the parent object above (a code can contain up to 10 levels).

[FIGURE 12 HERE]

Although MAXQDA and NVivo use functional coding scheme hierarchies that can act both as an organizational tool and retrieval engine (Lewins and Silver 2007:93), their hierarchical function does not preserve the relationship between the triplet elements, i.e., the relationships within the SVO template with S related to V and V to O. Instead, the hierarchy permits arranging codes in a manner that allows for a more practical approach to coding. For example, the pathway "Actor: Individual: Name of individual actor: Negro" is used in both "Participant-S" and "Participant-O". In these programs, the user would place a "Negro" code in the hierarchies of both "Participant-S" and "Participant-O" and the portions of text coded as "Negro" for the "Participant-S" pathway would remain separate from the portions coded as "Negro" under "Participant-O"[32].

CAQDAS Data Entry

Using the first *Atlanta Constitution* article (Excerpt 1), Figure 13 shows an ATLAS.ti screenshot for QNA coding.

[FIGURE 13 HERE]

Codes appear in the right window and the text to be coded on the left. CAQDAS coding is done by selecting a portion of the text in the left window and then assigning a code to the segment (e.g., the selected text to the code Semantic Triplet).[33]

For QNA each text segment from the newspaper article needs to be coded in a way that identifies its position in the story grammar. Consider semantic triplet 3: "A few weeks ago a house and warehouse were destroyed by the fire in Hinesville, and all the circumstances pointed to its being the work of an incendiary". "Incendiary" requires the code "Participant-S: Actor: Individual: Name of individual actor: incendiary" but this code then has to be assigned to the code family "Participant-S: Actor: Individual: Name of individual actor";[34] "Hinesville" requires a code named "Process: Simple process: Circumstances: Space: City: Hinesville" in the code family "Process: Simple process: Circumstances: Space: City".

CAQDAS Data Query

With the documents fully coded, data analysis can begin by using CAQDAS data queries. CAQDAS query commands are based on Boolean, semantic, and proximity operators. These commands can be used to retrieve data to answer QNA questions: WHO did WHAT? Pro or against WHOM? WHEN did they do it? WHERE? Consider the question: "WHAT did the "Negro" do?" The answer would involve constructing a query of the code family "Process: Simple Process: Verbal phrase" from "WITHIN" semantic triplets "ENCLOSING" the code "Participant-S: Actor: Individual: Name of individual actor: Negro". Figure 14 shows the list of

quotes extracted by the query: “confessed”, “implicated”, “stealing”, “broke into” and “set fire”.[35]

[FIGURE 14 HERE]

ATLAS.ti can retrieve information about the basic elements of a narrative (e.g., what an actor does) *if* (and only if) the values of a code (e.g., “Negro”) have been hard-coded in the code name. This is why the value a code takes *must* be embedded in the code name (“Negro” coded as “Participant-S: Actor: Individual: Name of individual actor: Negro”) and why this code *must* be arranged into family codes (“Participant-S: Actor: Individual: Name of individual actor”). So that such questions as “WHO confessed?” can be answered. *CAQDAS query tools dictate the code naming criteria.* Similarly, if the user needs to retrieve data about “WHO was acting,” for any specific type of action, the user must create a set of codes where the verbal phrase is hard-coded within the code name (e.g., “Process: Simple process: Verbal phrase: confessed”). Again, “confessed” would have to be coded as “Process: Simple process: Verbal phrase: confessed” to answer the questions about “WHO confessed?” and the code would have to be placed in the “Process: Simple process: Verbal phrase” code family so that it can be extracted by the query above for “WHAT did the “Negro” do?”

Document families can be used in conjunction with the query tool to set filters on a query to restrict query results. For instance, the filters set up for the query shown in Figure 15, will only return results for the documents included in that document family.

[FIGURE 15 HERE]

By coding the modifiers of the hierarchical objects of the story grammar (i.e., Macroevents and events) at the document level, CAQDAS users can use these elements of the story grammar to filter for subsets of data in the database. They can query, for instance, the subset of texts

reporting a certain type of event (e.g., hanging, shooting). Had not this information been coded at the document level, each and every query would require an extra set of query commands.[36] This would decrease both query efficiency (i.e., taking longer to run a query) and data reliability (i.e., increasing the likelihood of errors).

CAQDAS Data Query: Limitations

Querying in CAQDAS has its limitations: elements not explicitly present in a text cannot easily be coded and, therefore, cannot be queried. After all, if something is not there it cannot be selected and assigned to a code. Examples are the missing syntactic subject of passive clauses (e.g., "The Negro was carried into the woods.") or the missing syntactic object of clauses with intransitive verbs (e.g., the intransitive verb "strike" of a labor dispute has no syntactic object, but it does have a semantic implicit object, "employer"). Another example is the more general problem known as anaphora resolution. In the sentence, "He is said to have confessed the deed, and implicated several in the crime." The "he" refers to the lynched Negro, who, however, is nowhere to be found in the sentence (see Figure 16).

[FIGURE 16 HERE]

The user could select the "he" and assign it to the code "Participant-S: Actor: Individual: Name of individual actor: Negro" as the subject of the semantic triplet. However, when querying the data for the actors present in the story, the value "he" (rather than "Negro") would be returned. Depending upon how many "he" you have in the set of documents analyzed, the information provided by the query results could be meaningless: he, who? The Negro? The deputy Sheriff? Mr. Chapman? Again, text manipulation may be required (either before or after importing the documents to be coded) to make coding (and querying) more transparent. In this case, the user could edit the original text, substituting the word "Negro" to the word "he".

PC-ACE and CAQDAS Compared: A Summary

So, what are the differences between PC-ACE and CAQDAS? Does CAQDAS, a more familiar option than PC-ACE for social scientists, offer a viable solution to computer-assisted QNA? To answer these questions, let us bring together what we have learned about the main tasks involved in QNA: grammar setup, document cross-referencing, data entry, and data query.

Setting up the Grammar. PC-ACE provides all the tools to generate complex story grammars with relational and hierarchical objects. CAQDAS programs, while allowing the user to generate hierarchies or families of codes, are not designed to reproduce the multiple relations among the objects of complex story grammars.

Data entry. In PC-ACE, the source documents are external to the software (e.g., on a microfilm) and coding is based on manual data entry of information into text boxes or combo boxes. CAQDAS programs work with documents imported into the programs and coding is done by assigning codes to selected portions of a document. In CAQDAS, if documents are not available in digital form (a Word, PDF, or ASCII document), the texts will have to be converted to digital files (for instance, through scanning, and this can be a lengthy process). Problems of anaphora resolution may also require extensive editing of input documents in CAQDAS programs (which can also be a time consuming process).

Document cross-referencing. Each item of information coded in PC-ACE is cross-referred to its original source document. In data entry, users code information from different documents to construct a single, unified story (albeit, perhaps, with alternative story lines). In data query, information is (or can be) retrieved regardless of source documents. CAQDAS programs allow users to work on different source files but they do not provide simple tools to merge textual elements from different documents. CAQDAS programs provide an option for linking different

items of information within a single document and across documents (Lewins and Silver 2007: 63). These links, however, are hyperlinks that allow the user to jump from one segment of text to the linked text. While this is useful to remind the researcher of key intersections between documents, there is no way to retrieve this information in query form as in PC-ACE.[37] To overcome CAQDAS cross-referencing limitations, users can combine multiple documents into a single document or edit individual documents, but this is a labor intensive process.

Data query. PC-ACE and CAQDAS programs work very differently. While PC-ACE is an RDBMS program that relies on SQL (Structured Query Language) to extract information from coded data, CAQDAS programs provide their own set of query commands: Boolean, semantic, and proximity operators. Boolean operators only allow combinations of keywords and are the most common operators used in any information retrieval system. Semantic operators, (e.g., up, down and siblings in ATLAS.ti) can be used when codes are linked via so called transitive relations like "is part of", "is a", etc. something that defines a hierarchical relations. They extract information from codes having, for instance, "parent-children" relations or "sibling" relations. Proximity operators (e.g. within, encloses, overlapped by, overlaps, follows, precedes and co-occur in ATLAS.ti) describe spatial relations between coded textual elements.

Unfortunately, none of these CAQDAS tools allow the user to do what the SQL *where* command does in PC-ACE: set filters on the instances of specific coding categories (e.g., "Negro" as the specific value instance of the category <actor>). This means that a CAQDAS user interested in what a specific actor (e.g., "Negro") does, needs to "hard-code" the value ("Negro") into the code name (e.g., Participant-S: Actor: Individual: Name of individual actor: Negro) in order to be able to retrieve that information in a query. This approach is feasible with a limited number of actors and actions (more generally, of objects in the grammar). In large

projects, however, there may be thousands of distinct names of actors, actions, and all other objects, forcing CAQDAS users to work with literally thousands of codes. To overcome this limitation, doing QNA in CAQDAS programs would require 1. using a limited grammar, with a small number of simplex and a handful of complex objects; 2. working with codes that, for each simplex object, aggregate the distinct values in a handful of categories (e.g., for verbs, violence and communication, instead of "wound", "punch", "kick", "stone", "kill" or "say", "write", "tell", "publish"). Having coders perform both coding and aggregating at once may lead to serious reliability problems (Markoff et al., 1975: 37). Furthermore, this hard-coding of aggregate values also presumes that the researcher has a complete list of aggregate codes and of the individual values that make them up.

Another drawback of CAQDAS query systems concerns the computation of frequency counts. CAQDAS queries only extract quotations and frequency counts of codes. Like CAQDAS, PC-ACE does not have any statistical capabilities; but the SQL *count* statement in PC-ACE allows the researcher to compute frequency distributions for objects in relation to other objects (e.g., the frequency distribution of cities where mobs burned Negroes as opposed to hang them). Both software programs allow the user to export data in statistical packages for further manipulation but PC-ACE can also compute automatically network matrices that can be imported into network programs (e.g., UCINET, Pajek).

Conclusions

This paper has discussed a computer implementation of Quantitative Narrative Analysis (QNA) using PC-ACE (Program for Computer-Assisted Coding of Events) and CAQDAS programs (Computer-Assisted Qualitative Data Analysis Software), in particular, ATLAS.ti, MAXQDA, and NVivo. Quantitative narrative analysis is a powerful research tool for the quantitative

analysis of narrative texts. Given the complexity of the coding schemes (story grammars) upon which QNA relies, software is necessary to perform the analysis ("No software, no QNA"; Franzosi, 2010: 67).

PC-ACE was specifically designed to carry out QNA. CAQDAS programs were designed to help researchers in organizing and analyzing their material qualitatively (e.g., transcripts from focus groups or in-depth interviews). PC-ACE was born as a tool for quantitative analysis; CAQDAS programs as qualitative tools. Given these differences in basic epistemology and design strategy, a comparison between the two types of software may seem unfair and artfully set up. Yet, the popularity of CAQDAS programs may well make them the first port of call for any researcher interested in QNA; hence, our comparison. Certainly, and not unsurprisingly, PC-ACE provides a more comprehensive approach to quantitative narrative analysis. PC-ACE's power lies in its ability to preserve relationships between objects of the grammar and query these relationships to answer questions such as "who does what, pro/against whom, where and when." Both PC-ACE and CAQDAS allow users to export tables' content and query results to other software packages for specialized analyses. Contrary to CAQDAS, however, PC-ACE computes automatically the network matrix that can be cut and pasted into network analysis software. Indeed, the application of QNA in PC-ACE has helped to shed light on such socio-historical puzzles as the rise of Italian fascism (1919-1922) on the basis of over 50,000 newspaper articles from three different newspapers yielding some 200,000 semantic triplets (Franzosi, 1997, 2010: 107-142, 2011) and the lynching of African Americans in Georgia (1875-1930) on the basis of over 1,300 newspaper articles for nearly 7,000 triplets (Franzosi, in press, Franzosi et al., 2011).[38]

CAQDAS programs have limitations that affect 1. the complexity of the story grammar that can be used; 2. the scale of the project that can be carried out; 3. and the range of questions that can be asked. It is the query tools of CAQDAS programs that set the main obstacle to a full QNA implementation. Contrary to PC-ACE, where SQL-based queries (Structured Query Language) allow users to query the data across a number of coding categories and for any value of any category (e.g., who did what in the city of Hinesville), CAQDAS query tools were designed to extract units of text assigned to various codes across different documents (users can also extract frequencies of these codes). Users cannot set filters for specific values on the objects involved in the query (e.g., the actions performed by the specific actor: "Negro"). To perform queries involving specific values of a code (again, the value "Negro" of the code "Name of individual actor"), these values must be hard coded into the name of the code (e.g., Semantic triplet: Participant-S: Actor: Individual: Name of individual actor: Negro). Even for small projects, based on a limited number of documents, these limitations could lead to a bewildering number of codes – in the hundreds if not in the thousands, way too many for any sensible approach to coding.[39]

As a result, researchers wishing to use CAQDAS programs for QNA may need to focus on a handful of specific objects in relation to other objects (e.g., where an action occur) and on a handful of specific values of objects (e.g., "Negro," "police", "woman" as the value of an actor).[40] They may also need to work with a simplified grammar based on highly aggregated codes.

For those researchers familiar with any of the CAQDAS programs, reluctant to invest in the learning of specialized software as PC-ACE, and willing to accept the limitations in project's

scope and questions, a limited implementation of QNA may be all they need. And this paper has shown how to do just that: implement a limited QNA in CAQDAS programs.

Acknowledgments

We are grateful to Susanne Friese of ATLAS.ti for her continuous help over the years on how to implement QNA in ATLAS.ti and for her comments on a draft of this paper. We are also grateful to Emory University (and the University Research Committee) for its support of a project on lynching in Georgia (1875-1930) and to Professor Woody Beck for providing the references to the 1300 newspaper articles that narrate the Georgia lynchings.

References

Abell, Peter. 2004. "Narrative Explanation: An Alternative to Variable Centered Explanation?" *Annual Review of Sociology*, Vol. 30, pp. 287-310.

Alexa, Melina and Cornelia Zuell. 2000. "Text Analysis Software: Commonalities, Differences and Limitations: The Results of a Review." *Quality & Quantity*, Vol. 34, No. 3, pp. 299-321.

Beck, Woody and Stewart Tolnay. 1990. "The Killing Fields of the Deep South: The Market for Cotton and the Lynching of Blacks, 1882-1930." *American Sociological Review* 55:526-539.

Carley, Kathleen. 1993, "Coding Choices for Textual Analysis: A Comparison of Content Analysis and Map Analysis" *Sociological Methodology*, Vol. 23, pp. 75-126

De Fazio, Gianluca. 2011. *Political Radicalization in the Making: the Civil Rights Movement in Northern Ireland, 1968-1972.* Doctoral Dissertation. Sociology Department. Emory University.

Doyle, Sophie. 2009. *"Stories Most of All": A Content Analysis of Bestselling Children's Books in Britain (1964-2004)*. Doctoral Dissertation. Sociology Department. University of Oxford.

Friese, Susanne. 2011. "Using ATLAS.ti for Analyzing the Financial Crisis Data."*Forum Qualitative Sozialforschung / Forum: Qualitative Social Research*, Vol. 12, No. 1, http://nbn-resolving.de/urn:nbn:de:0114-fqs1101397.

Franzosi, Roberto. 2004. *From Words to Numbers: Narrative, Data, and Social Science*. Cambridge: Cambridge University Press.

2008. *Content Analysis*. Benchmarks in Social Research Methods series (Quantitative Applications in the Social Sciences). 4 vols. Thousand Oaks, CA: Sage.

2010. *Quantitative Narrative Analysis*. Quantitative Applications in the Social Sciences Series. Thousand Oaks, CA: Sage.

In press. "On Quantitative Narrative Analysis." In: *Varieties of Narrative Analysis*, edited by James A. Holstein and Jaber F. Gubrium. Los Angeles: Sage.

2011. "Narratives and Networks: Mapping Social Relations during Revolutionary Periods (Italy, 1919-1922)." Unpublished manuscript, currently under journal review.

Franzosi, Roberto, Gianluca De Fazio, and Stefania Vicari. 2011. "Ways of Measuring Agency: An Application of Quantitative Narrative Analysis to Lynchings in Jim Crow South." Unpublished manuscript, currently under journal review.

Halliday, M.A.K. 1994 (1985). *An Introduction to Functional Grammar*. London: Arnold.

Junker, Andrew. 2012. *The Persistence of Gods and Emperors: The Chinese Democracy Movement and the Falun Gong*. Doctoral Dissertation. Sociology Department. Yale University.

Lewins, Ann and Christina Silver. 2007. *Using Software in Qualitative Research: A Step-by-Step Guide*. London: Sage.

Markoff, John, Gilbert Shapiro, and Sasha Weitman. 1975. "Toward the Integration of Content Analysis and General Methodology", *Sociological Methodology*, Pp. 1-58.

Podolny, Joel M. and Marya Hill-Popper. 2004. "Hedonic and Transcendent Conceptions of Value", *Industrial and Corporate Change,* Vol. 13 (1): 91-116.

Popping, Roel. 2000. *Computer-Assisted Text Analysis*. Thousand Oaks, CA: Sage.

Vicari, Stefania. 2008. *Contemporary Protest: The Online Framing of Local and Global Dynamics*. Doctoral Dissertation. Sociology Department. University of Reading.

Appendix I

Savannah, Feb 8. A few weeks ago a house and warehouse were destroyed by the fire in Hinesville, and all the circumstances pointed to its being the work of an incendiary. The people have been greatly wrought up in consequence. Intelligence received here tonight states that a Negro was arrested there yesterday on the charge of burning the houses aforesaid. He is said to have confessed the deed, and implicated several in the crime. After a preliminary investigation he was committed to jail in Hinesville. Last night a band of armed men overpowered the deputy Sheriff, who had the prisoner in charge, and carrying him off to the woods, shot him to death. Great excitement prevails in that section.

Excerpt 1 - *Atlanta Constitution* article (February 9, 1888)

Savannah, Ga, February 9. Very little information can be obtained from Liberty county about the lynching of the Negro incendiary Tuesday night. About a fortnight ago The Constitution published an account of a fire at Johnson's station, on the Savannah, Florida and Western railway. Mr. Chapman lost a store and the railway company's warehouse was burned, along with several other buildings. It was suspected that the fire was started by an incendiary, and on Tuesday a Negro was arrested on suspicion. He was given a preliminary hearing and confessed that he was one of a party of five who broke into Mr. Chapman's store. After stealing all they could carry off, the burglars sprinkled kerosene about the building and set fire to it. The magistrate committed the Negro to jail. While the deputy Sheriff was on his way to the Hinesville jail, he was surprised by a crowd of fifteen men, who took the prisoner away from him. The Negro was carried into the woods, and it is supposed he was hung or burned. The officer never saw him more. It is expected that the other incendiaries will share the same fate.

Excerpt 2 - *Atlanta Constitution* article (February 10, 1888)

A few weeks ago a house and warehouse were destroyed by the fire in Hinesville at *Johnson's station, on the Savannah, Florida and Western railway*, and all the circumstances pointed to its being the work of an incendiary. *Mr. Chapman lost a store and the railway company's warehouse was burned, along with several other buildings*. The people have been greatly wrought up in consequence. Intelligence received here tonight states that a Negro was arrested there yesterday *on Tuesday* on the charge of burning the houses aforesaid. He is said to have confessed the deed, and implicated several in the crime, *that he was one of a party of five who broke into Mr. Chapman's store*. *After stealing all they could carry off*, *the burglars sprinkled kerosene about the building and set fire to it*. After a preliminary investigation he was committed to jail in Hinesville. Last night *on Tuesday while the deputy Sheriff was on his way to the Hinesville jail* a band of *fifteen* armed men overpowered the deputy Sheriff, who had the prisoner in charge, and carrying him off to the woods, shot him to death. *It is supposed he was hung or burned*. Great excitement prevails in that section.

Excerpt 3 - *Atlanta Constitution* combined articles (February 9 and 10, 1888)

Appendix II

Legend:

--> Rewrite Rule (or Production): the object to the left of the arrow is "rewritten" in terms of the object(s) to the right;
< > indicates that an object can be rewritten (i.e., is not a terminal element of the grammar);
[ ] indicates that an object is optional;
{ } indicates that an object may have multiple instances;
| indicates a logical OR between alternative values that a simplex can take;
++ denotes One-To-Many (Hierarchical) complex objects (i.e., objects made up of simplex and complex objects);
+ denotes One-To-Few complex objects (objects enclosed in <> without a + or ++ are simplex objects);
(1a)(1b)... denote grouped and mutually exclusive objects.

Newspaper article --> <Newspaper name>
<Newspaper date>
<Page number>

<++Macroevent> --> {<Victim>} [{<++Event>}]

<++Event> --> <Type of event> [<+Alternative event>] [{<++Semantic Triplet>}]

<+Semantic Triplet> --> <+Participant-S> <+Process> [{<+Participant-O>}]
[<Relation to next triplet>] [<+Alternative triplet>]

30:
31: <+Participant-S> --> {<+Actor>}
32:
33: <+Actor> --> {<+Individual (1a)>} {<+Collective actor (1b)>} [<+Organization (1c)>]
34:
35: <+Individual> --> <Name of individual actor> [{<+Personal characteristics>}] <Aggregate actor>
36: <Name of individual actor> --> ? | deputy Sheriff | farmer | girl | incendiary | ...
37:
38: <+Personal characteristics> --> [<+First name and last name>] [<Gender>] [<+Age>] [<Race>]
39: [{<+Family relationship>}] [<+Residence>] [{<Nationality>}]
40: [{<Body part>}] [{<Type of actor (Adjective)>}]
41: [<Party affiliation: Political party>]
42:
43: <+First name and last name> --> [<First name>] [<Middle name: First name>] <Last name>
44: <First name> --> John | Mary | Michael | Mose | ...
45: <Middle name: First name> --> John | Mary | Michael | Mose | ...
46: <Last name> --> Chapman | Fisher | Herring | Larson | ...
47: <Gender> --> female | male
48:
49: <+Age> --> <Qualitative age (1a)> <Exact age: Numeric value (1b)>
50: <Qualitative age> --> old | young | ...
51: <Exact age: Numeric value> --> 4 | 6 | 15 | 20 | 30 | ...
52: <Race> --> Negro | white | ...
53:
54: <+Family relationship> --> <Type of relationship> <+Actor>
55: <Type of relationship> --> cousin | daughter | father | mother | son | ...
56: <+Actor> --> Rewrite rules for this object on line 33
57:
58: <+Residence> --> <+Space>
59:
60: <+Space> --> {<+City (1a)>} {<+Territory (1b)>}
61:
62: <+City> --> [<Space qualifier>] [<Spatial direction>] [<+Distance from city>]

```
63:          [<+Locality within city>] [<Locality near city>] <City name> [<County>] [<State>]
64:          [{<+Relation to other location>}]
65:      <Space qualifier> --> along | behind | close to | in front of | near | ...
66:      <Spatial direction> --> across | to | towards | ...
67:
68:      <+Distance from city> --> [<Approximate qualifier>] <Numeric value> <space unit>
69:                   [<Direction: Cardinal/Ordinal>]
70:        <Approximate qualifier> --> about | circa | ...
71:        <Numeric value> --> 4 | 6 | 15 | 20 | 30 | ...
72:        <space unit> --> foot | mile | yard | miles | ...
73:        <Direction: Cardinal/Ordinal> --> East | North | Northeast | Northwest | South
74:
75:      <+Locality within city> --> {<+Address (1a)>} {<+Building (1b)>} {<Neighborhood (1c)>}
76:
77:        <+Address> --> [<Number: Numeric value>] <Street (1a)> <Square (1b)> [<Neighborhood>]
78:          <Number: Numeric value> --> 4 | 6 | 15 | 20 | 30 | ...
79:          <Street> --> Clifton Road | ...
80:          <Square> --> String
81:          <Neighborhood> --> String
82:
83:        <+Building> --> [<Headquarters of>] [<Proper name of building>] <Type of building>
84:                [<+Address>]
85:          <Headquarters of> --> String
86:          <Proper name of building> --> Hinesville jail | ...
87:          <Type of building> --> bank | jail | store | warehouse | ...
88:          <+Address> --> Rewrite rules for this object on line 77
89:        <Neighborhood> --> String
90:      <Locality near city> --> beach | lake | woods | ...
91:      <City name> --> Athens | Atlanta | Blackshear | Hinesville | Savannah | ...
92:      <County> --> Liberty | Pierce | ...
93:      <State> --> Georgia | South Carolina | ...
94:
95:      <+Relation to other location> --> [<+Distance from Location>] <Direction: Cardinal/Ordinal>
```

96: <+Location>
97:
98: <+Distance from Location> --> [<Approximate qualifier>] <Numeric value> <Space unit>
99: <Approximate qualifier> --> about | circa | ...
100: <Numeric value> --> 4 | 6 | 15 | 20 | 30 | ...
101: <Space unit> --> foot | mile | yard | miles | ...
102: <Direction: Cardinal/Ordinal> --> East | North | Northeast | Northwest | South
103:
104: <+Location> --> <City: City name (1c)> <County (1b)> <State (1a)>
105: <City: City name> --> Athens | Atlanta | Blackshear | Hinesville | Savannah | ...
106: <County> --> Liberty | Pierce | ...
107: <State> --> Georgia | South Carolina | ...
108:
109: <+Territory> --> [<+Distance from territory>] [<Spatial direction>] <+Type of territory>
110: [<Non administrative unit>] [{<+Relation to other location>}]
111:
112: <+Distance from territory> --> [<Approximate qualifier>] <Numeric value> <space unit>
113: <Approximate qualifier> --> about | circa | ...
114: <Numeric value> --> 4 | 6 | 15 | 20 | 30 | ...
115: <space unit> --> foot | mile | yard | miles | ...
116: <Spatial direction> --> across | to | towards | ...
117:
118: <+Type of territory> --> <County (1b)> <State (1a)>
119: <County> --> Liberty | Pierce | ...
120: <State> --> Georgia | South Carolina | ...
121: <Non administrative unit> --> Deep South | Far West | Jim Crow South | ...
122: <+Relation to other location> --> Rewrite rules for this object on line 95
123: <Nationality> --> American | Irish | Mexican | ...
124: <Body part> --> arm | hand | head | leg | ...
125: <Type of actor (Adjective)> --> alive | armed | helpless | highly respected | ...
126: <Party affiliation: Political party> --> Democratic | Republican | ...
127: <Aggregate actor> --> mob | workers | ...
128:

129: <+Collective actor> --> <Name of collective actor> [{<+Collective characteristics>}] <Aggregate actor>
130: <Name of collective actor> --> band | individuals | party | posse | mob | ...
131:
132: <+Collective characteristics> --> [{<Gender>}] [{<+Age>}] [{<Race>}] [{<+Family relationship>}]
133: [{<+Residence>}] [{<Nationality>}] [{<Type of actor (Adjective)>}]
134: [{<Job>}] [{<Party affiliation: Political party>}]
135: [{<+Group composition>}]
136: [{<+Subgroup (among which): Subset (among which)>}] [<+Number>]
137: <Gender> --> female | male
138: <+Age> --> Rewrite rules for this object on line 49
139: <Race> --> Negro | white | ...
140: <+Family relationship> --> Rewrite rules for this object on line 54
141: <+Residence> --> Rewrite rules for this object on line 58
142: <Nationality> --> American | Irish | Mexican | ...
143: <Type of actor (Adjective)> --> alive | armed | helpless | highly respected | ...
144: <Job> --> banker | butcher | doctor | farmer | peasant | ...
145: <Party affiliation: Political party> --> Democratic | Republican | ...
146:
147: <+Group composition> --> <Part qualifier> <+Actor>
148: <Part qualifier> --> among which | ...
149: <+Actor> --> Rewrite rules for this object on line 33
150:
151: <+Subset (among which)> --> <Part qualifier> <+Actor>
152: <Part qualifier> --> among which | ...
153: <+Actor> --> Rewrite rules for this object on line 33
154:
155: <+Number> --> [<Comparative qualifier>] [<Approximate qualifier>] <+Qualitative value (1a)>
156: <+Quantitative value (1b)>
157: <Comparative qualifier> --> fewer | less | more | ...
158: <Approximate qualifier> --> about | circa | ...
159:
160: <+Qualitative value> --> [<Quantitative qualifier>] <Numeral>
161: <Quantitative qualifier> --> a few | little | much | ...

```
162:                 <Numeral> --> many | several | ...
163:
164:               <+Quantitative value> --> <Numeric value (1a)> <+Range of values (1b)> [<+Value out of total>]
165:                 <Numeric value> --> 4 | 6 | 15 | 20 | 30 | ...
166:
167:                 <+Range of values> --> <Lower value: Numeric value> <Upper value: Numeric value>
168:                   <Lower value: Numeric value> --> 4 | 6 | 15 | 20 | 30 | ...
169:                   <Upper value: Numeric value> --> 4 | 6 | 15 | 20 | 30 | ...
170:
171:                 <+Value out of total> --> <Out of total: Numeric value> <Numeric value>
172:                   <Out of total: Numeric value> --> 4 | 6 | 15 | 20 | 30 | ...
173:                   <Numeric value> --> 4 | 6 | 15 | 20 | 30 | ...
174:           <Aggregate actor> --> mob | workers | ...
175:
176:         <+Organization> --> <Name of organization> [<+Type of organization>] <Aggregate actor>
177:                         [<+Number and level of organizational unit>] [<+Name of unit>]
178:                         [<+Number of individuals in unit>] [{<+Locality of unit>}] [{<+Ownership>}]
179:           <Name of organization> --> Savannah, Florida and Western railway | ...
180:
181:           <+Type of organization> --> <Economic organization (1a)> <State organization (1b)>
182:                                 <Political party (1c)> <Other organization (1d)>
183:             <Economic organization> --> General Motors | United Fruit Company | ...
184:             <State organization> --> police | ...
185:             <Political party> --> Democratic | Republican | ...
186:             <Other organization> --> Baptist Church | KKK | NAACP | ...
187:           <Aggregate actor> --> mob | workers | ...
188:
189:           <+Number and level of organizational unit> --> <+Number of units: Number> <Level of unit>
190:             <+Number> --> Rewrite rules for this object on line 155
191:             <Level of unit> --> office | sector | ...
192:
193:           <+Name of unit> --> <Group/Holding (1a)> <Division (1b)> <Plant (1d)> <Department (1c)> <Office (1e)>
194:             <Group/Holding> --> String
```

```
195:                <Division> --> public security | ...
196:                <Plant> --> String
197:                <Department> --> fire | police | ...
198:                <Office> --> administrative | central | ...
199:
200:              <+Number of individuals in unit> --> {<+Number>}
201:                <+Number> --> Rewrite rules for this object on line 155
202:
203:              <+Locality of unit> --> {<+Space>}
204:                <+Space> --> Rewrite rules for this object on line 60
205:
206:              <+Ownership> --> {<+Individual (1a)>}
207:                <+Individual> --> Rewrite rules for this object on line 35
208:
209:          <+Process> --> {<+Simple process (1a)>} {<+Complex process (1b)>}
210:
211:            <+Simple process> --> <Verbal phrase> <Aggregate action> [<Negation>] [<Modal verb>] [<+Circumstances>]
212:              <Verbal phrase> --> arrested | assaulted | broke into | burned | carried | ...
213:              <Aggregate action> --> accusation | agreement | apprehension | assembling | communication | ...
214:              <Negation> --> no | not | ...
215:              <Modal verb> --> can | could | have to | must | shall | ...
216:
217:              <+Circumstances> --> [{<Action type (Adverb)>}] {<+Time>} [{<+Duration>}] {<+Space>} [{<+Reason>}]
218:                                   [{<+Number>}] [{<+Instrument>}] [{<+Outcome>}] [{<+Content>}]
219:                <Action type (Adverb)> --> allegedly | preliminarly | quickly | ...
220:
221:                <+Time> --> [<Approximate qualifier>] <+Date> [<+Time of day>] [<Temporal periodicity>]
222:                  <Approximate qualifier> --> about | circa | ...
223:
224:                  <+Date> --> <+Definite date (1a)> <+Indefinite date (1b)>
225:
226:                    <+Definite date> --> [<Temporal direction>] <Definite date>
227:                      <Temporal direction> --> after | ago | before | ...
```

228: <Definite date> --> 02/08/1888 | 02/09/1888 | 06/16/1894 | ...
229:
230: <+Indefinite date> --> [<Temporal direction>] [<Time qualifier>] [<+Time expression>]
231: <+Reference yardstick>
232: <Temporal direction> --> after | ago | before | ...
233: <Time qualifier> --> early | late | mid | ...
234:
235: <+Time expression> --> <Day (1a)> <Month (1b)> <Season (1c)> [<Quantitative qualifier>]
236: <Generic temporal expression (1d)>
237: <Day> --> Friday | Monday | Saturday | Sunday | Thursday
238: <Month> --> April | August | December | February | January
239: <Season> --> Autumn | Fall | Spring | Summer | Winter
240: <Quantitative qualifier> --> a few | little | much | ...
241: <Generic temporal expression> --> days | months | weeks | years | yesterday | ...
242:
243: <+Reference yardstick> --> <Article date: Definite date (1a)> <Newspaper date (1b)>
244: <+Semantic Triplet (1c)> <+Event (1d)> <+Macroevent (1e)>
245: <Article date: Definite date> --> 02/08/1888 | 02/09/1888 | 06/16/1894 | ...
246: <Newspaper date> --> 02/09/1888 | 02/10/1888 | 06/16/1894 | ...
247: <+Semantic Triplet> --> Rewrite rules for this object on line 26
248: <++Event> --> Rewrite rules for this object on line 22
249: <++Macroevent> --> Rewrite rules for this object on line 18
250:
251: <+Time of day> --> [<Approximate qualifier>] [<Time qualifier>] <+Exat Hour: Exact Hour (1a)>
252: <+Indefinite time of day (1b)>
253: <Approximate qualifier> --> about | circa | ...
254: <Time qualifier> --> early | late | mid | ...
255:
256: <+Exact Hour> --> <Hour and minute>
257: <Hour and minute> --> 12:00:00 AM | ...
258:
259: <+Indefinite time of day> --> [<Time qualifier>] <Moment of the day>
260: <Time qualifier> --> early | late | mid | ...

261: <Moment of the day> --> afternoon | dawn | morning | night | ...
262: <Temporal periodicity> --> daily | monthly | weekly | yearly | ...
263:
264: <+Duration> --> <Qualitative Duration (1a)> <+Quantitative Duration (1b)>
265: <Qualitative Duration> --> awhile | ...
266:
267: <+Quantitative Duration> --> {<+Number>} {<Time unit>}
268: <+Number> --> Rewrite rules for this object on line 155
269: <Time unit> --> hours | minutes | seconds | ...
270: <+Space> --> Rewrite rules for this object on line 60
271:
272: <+Reason> --> <Name of reason> [<+Semantic Triplet (1a)>] [<+Event (1b)>] [<+Macroevent (1c)>]
273: <Name of reason> --> assault | charge | lynching | murder | ...
274: <+Semantic Triplet> --> Rewrite rules for this object on line 26
275: <++Event> --> Rewrite rules for this object on line 22
276: <++Macroevent> --> Rewrite rules for this object on line 18
277: <+Number> --> Rewrite rules for this object on line 155
278:
279: <+Instrument> --> [{<+Number>}]
280: <+Number> --> Rewrite rules for this object on line 155
281:
282: <+Outcome> --> <Name of outcome> [{<+Quantification of outcome>}] [<+Semantic Triplet (1a)>]
283: [<+Event (1b)>] [<+Macroevent (1c)>]
284: <Name of outcome> --> death | ...
285:
286: <+Quantification of outcome> --> [<+Number>]
287: <+Number> --> Rewrite rules for this object on line 155
288: <+Semantic Triplet> --> Rewrite rules for this object on line 26
289: <++Event> --> Rewrite rules for this object on line 22
290: <++Macroevent> --> Rewrite rules for this object on line 18
291:
292: <+Content> --> <Name of content> [<+Semantic Triplet (1a)>] [<+Event (1b)>] [<+Macroevent (1c)>]
293: <Name of content> --> charge | deed | speech | ...

```
294:                <+Semantic Triplet> --> Rewrite rules for this object on line 26
295:                <++Event> --> Rewrite rules for this object on line 22
296:                <++Macroevent> --> Rewrite rules for this object on line 18
297:
298:          <+Complex process> --> <+Simple process> <+Other process>
299:            <+Simple process> --> Rewrite rules for this object on line 211
300:
301:            <+Other process> --> <+Simple process (1a)> <+Nominalization (1b)>
302:              <+Simple process> --> Rewrite rules for this object on line 211
303:
304:              <+Nominalization> --> <Verbal noun> <Aggregate action> [{<+Time>}] [<Negation>] [{<+Space>}]
305:                                    [{<Action type (Adverb)>}] [{<+Reason>}] [{<+Outcome>}] [{<+Instrument>}]
306:                                    [{<+Number>}] [{<+Content>}]
307:                <Verbal noun> --> assembly | gathering | meeting | ...
308:                <Aggregate action> --> accusation | agreement | apprehension | assembling | communication | ...
309:                <+Time> --> Rewrite rules for this object on line 221
310:                <Negation> --> no | not | ...
311:                <+Space> --> Rewrite rules for this object on line 60
312:                <Action type (Adverb)> --> allegedly | preliminarly | quickly | ...
313:                <+Reason> --> Rewrite rules for this object on line 272
314:                <+Outcome> --> Rewrite rules for this object on line 282
315:                <+Instrument> --> Rewrite rules for this object on line 279
316:                <+Number> --> Rewrite rules for this object on line 155
317:                <+Content> --> Rewrite rules for this object on line 292
318:
319:        <+Participant-O> --> [<Case>] <+Actor (1a)> <+Physical object (1b)> <+Abstract object (1c)>
320:          <Case> --> across | from | to | ...
321:          <+Actor> --> Rewrite rules for this object on line 33
322:
323:          <+Physical object> --> [<Proper name>] [{<+Number>}] [<+Ownership>] <+Implicit object>
324:            <Proper name> --> String
325:            <+Number> --> Rewrite rules for this object on line 155
326:            <+Ownership> --> Rewrite rules for this object on line 206
```

```
327:
328:                <+Implicit object> --> {<+Actor>}
329:                  <+Actor> --> Rewrite rules for this object on line 33
330:
331:              <+Abstract object> --> <Name of abstract object> <+Implicit object>
332:                <Name of abstract object> --> fear | ...
333:                <+Implicit object> --> Rewrite rules for this object on line 328
334:            <Relation to next triplet> --> because | therefore | when | ...
335:
336:            <+Alternative triplet> --> <Alternative description> <+Semantic Triplet>
337:              <Alternative description> --> True/False
338:              <+Semantic Triplet> --> Rewrite rules for this object on line 26
339:          <Type of event> --> arrest | assault | burglary | fire | Investigation | ...
340:
341:          <+Alternative event> --> <Alternative description> <+Event>
342:            <Alternative description> --> True/False
343:            <++Event> --> Rewrite rules for this object on line 22
344:
345:          <+Semantic Triplet> --> Rewrite rules for this object on line 26
346:        <Victim> --> unnamed Negro | ...
347:
348:        <++Event> --> Rewrite rules for this object on line 22
```

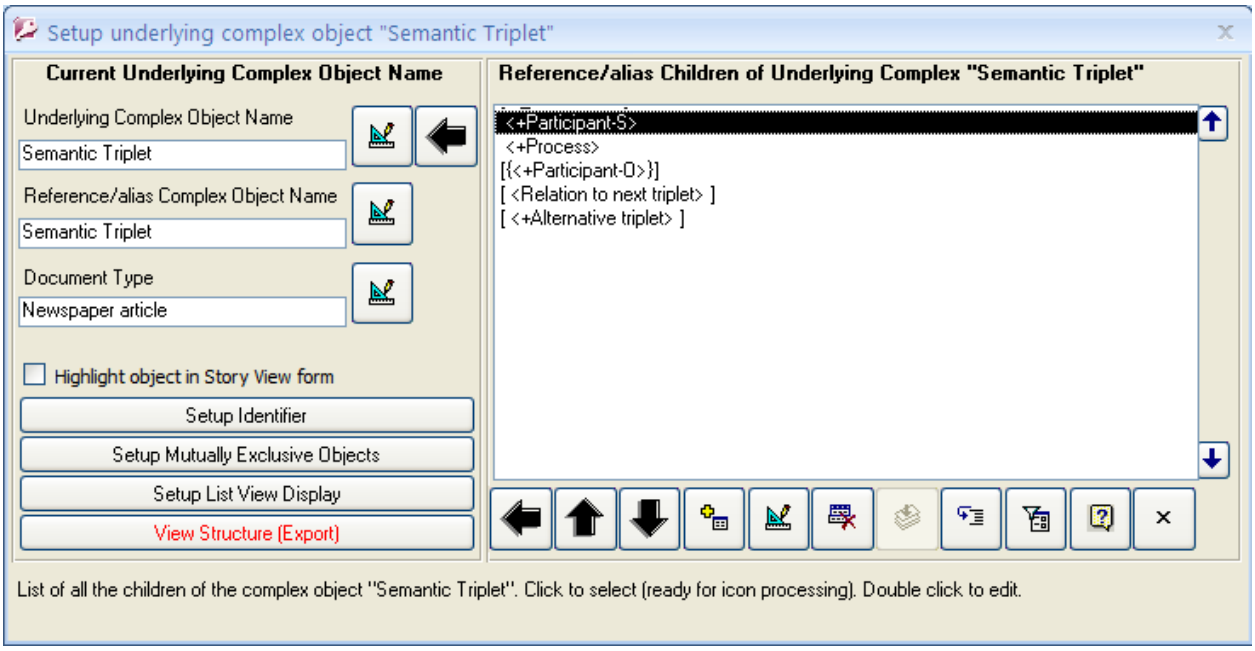

Figure 1: PC-ACE Setup Form showing the Children Objects of the Complex Object Semantic Triplet

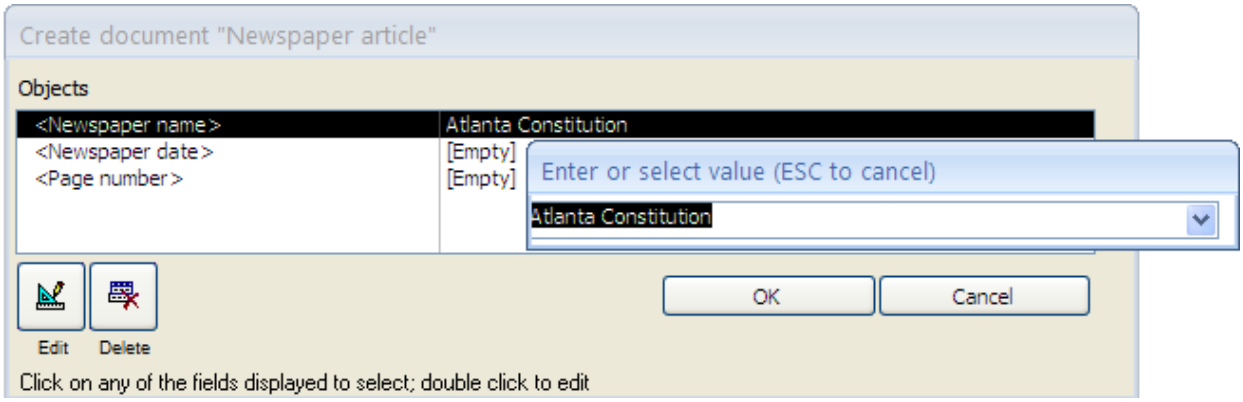

Figure 2: PC-ACE Create Document Form

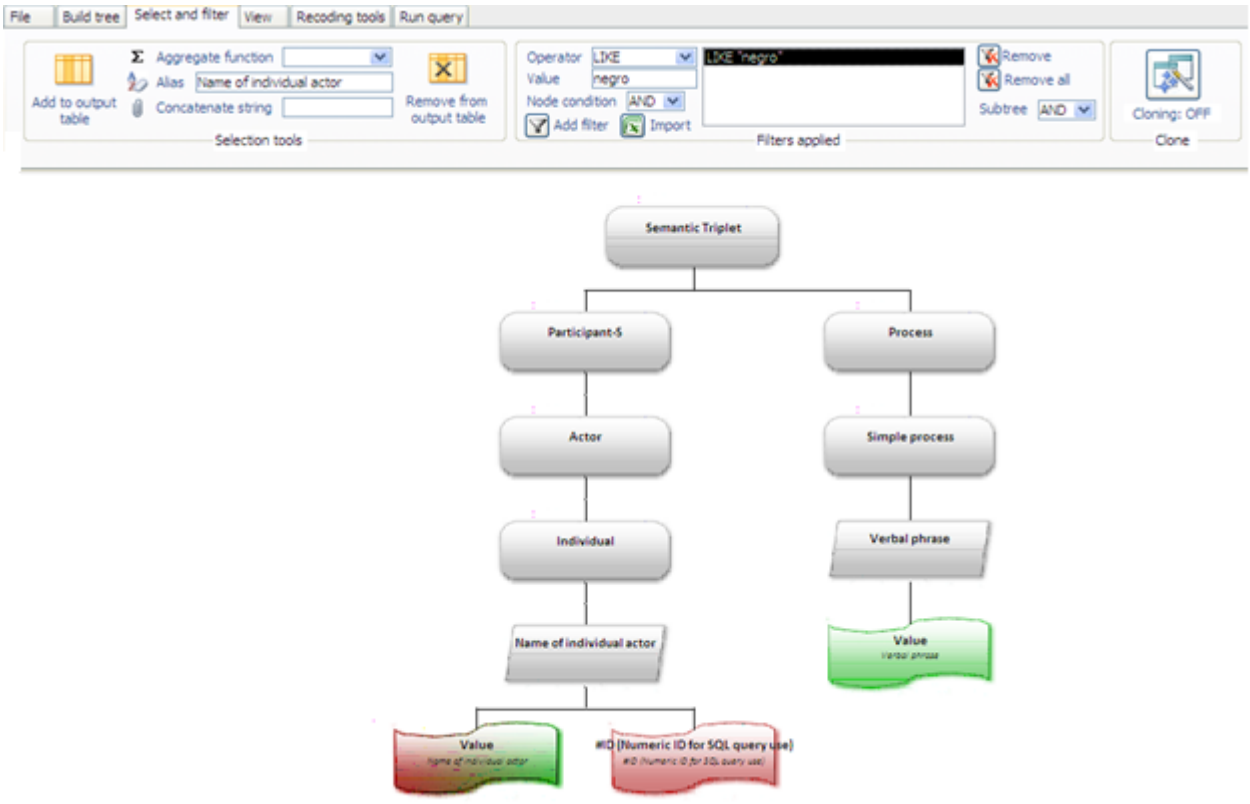

Figure 3: PC-ACE Query Manager: Querying for Negro's Actions

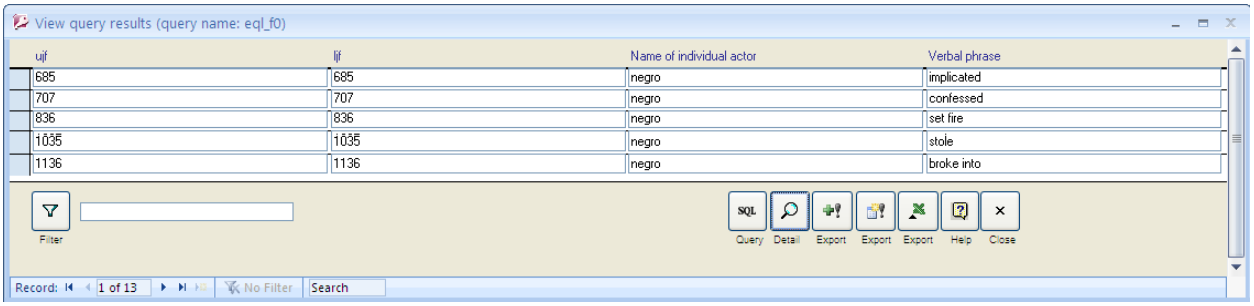

Figure 4: PC-ACE Query Manager: Query Results (Negro's Actions)

Aggregate code	FREQUENCY
Violence against people	1190
Movement	991
Force/coerce	565
Control	324
Search	315
Communication	275
Doing	267
Facilitation/help	180
Law	146
Apprehension	140
Request	121
Other	109
Senses	109
Assembling	103

Table 1: The What Revisited: Aggregated Actions in the Lynching Database (Frequency >100)

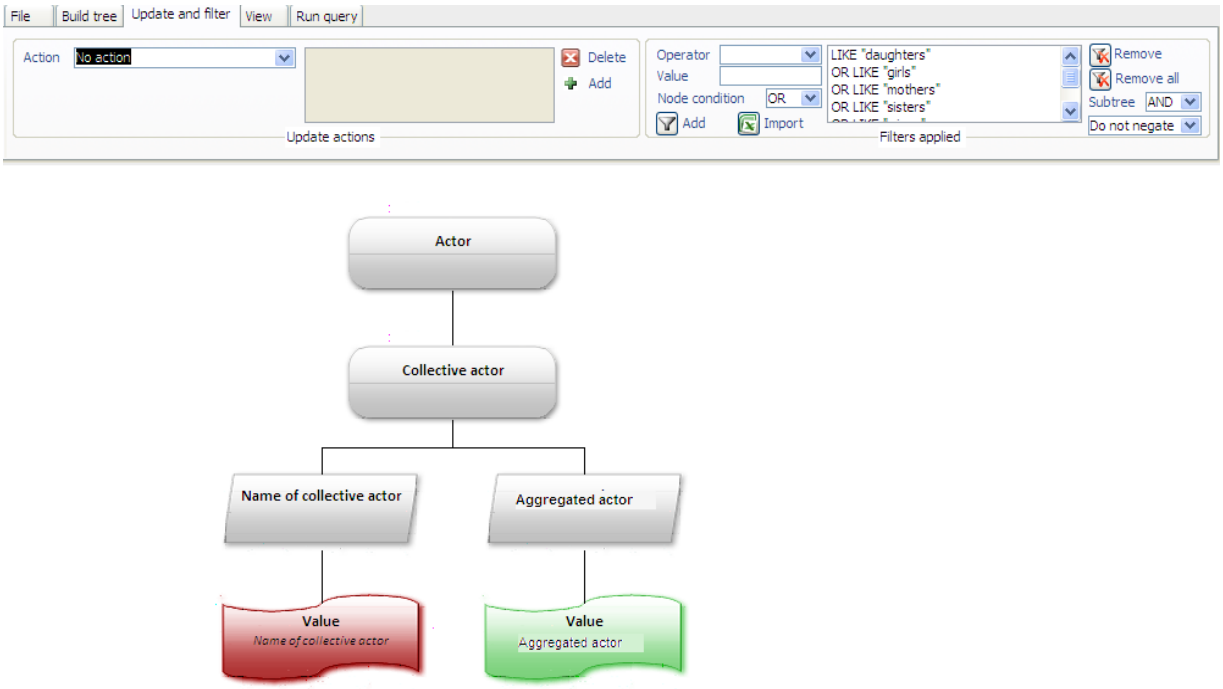

Figure 5: PC-ACE Update Manager: Setting up a Filter on the Code Name of Collective Actor

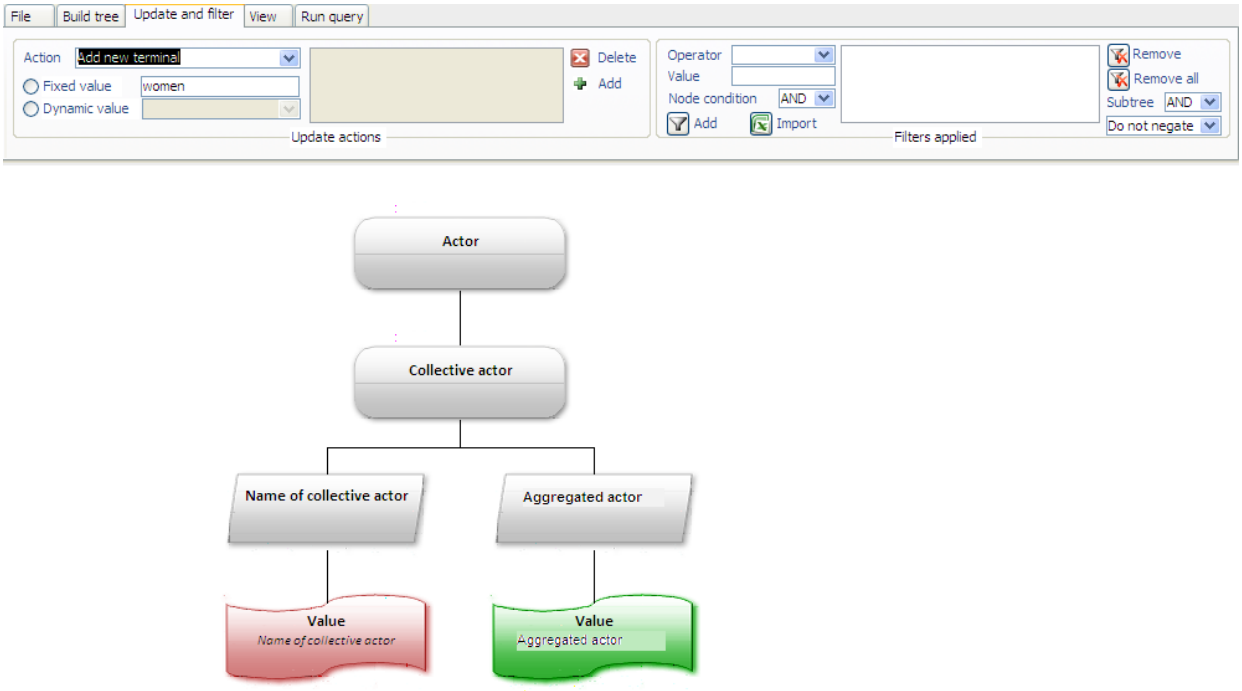

Figure 6: PC-ACE Update Manager: Update Aggregate Category for Collective Actor

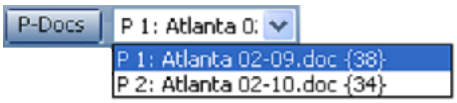

Figure 7: Assigning Articles to Hermeneutic Units in ATLAS.ti

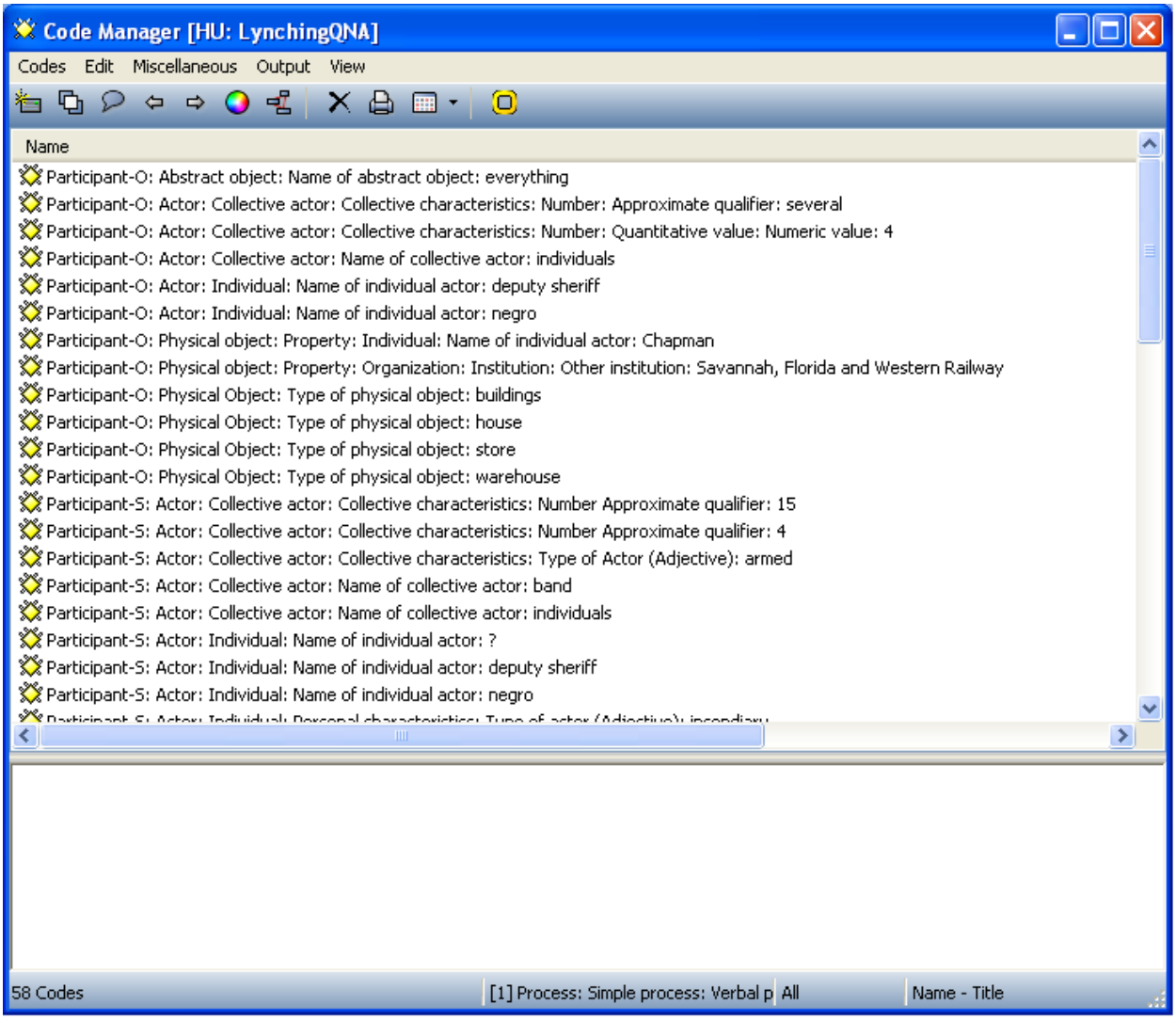

Figure 8: ATLAS.ti Code Manager

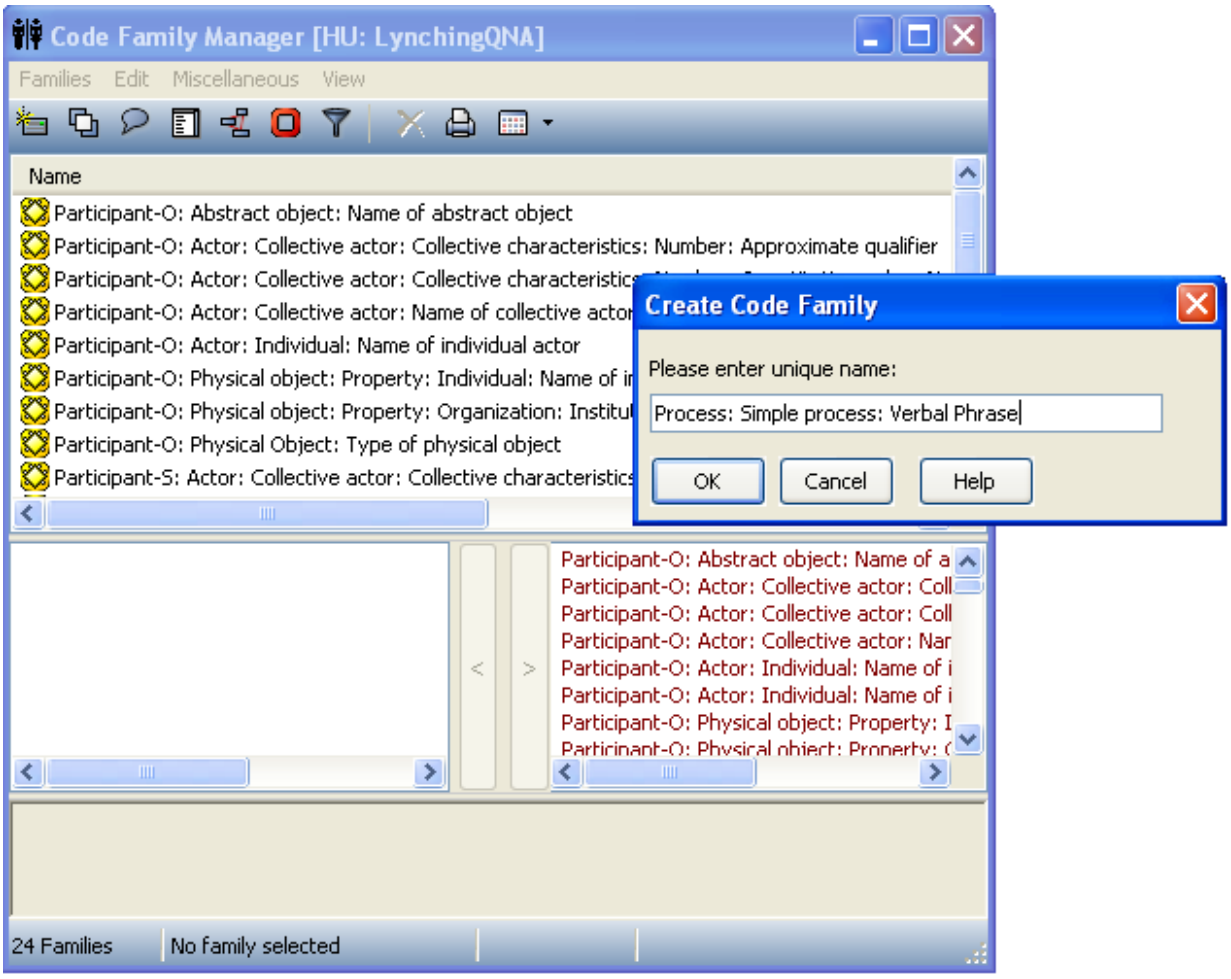

Figure 9: Creating a Code Family in ATLAS.ti

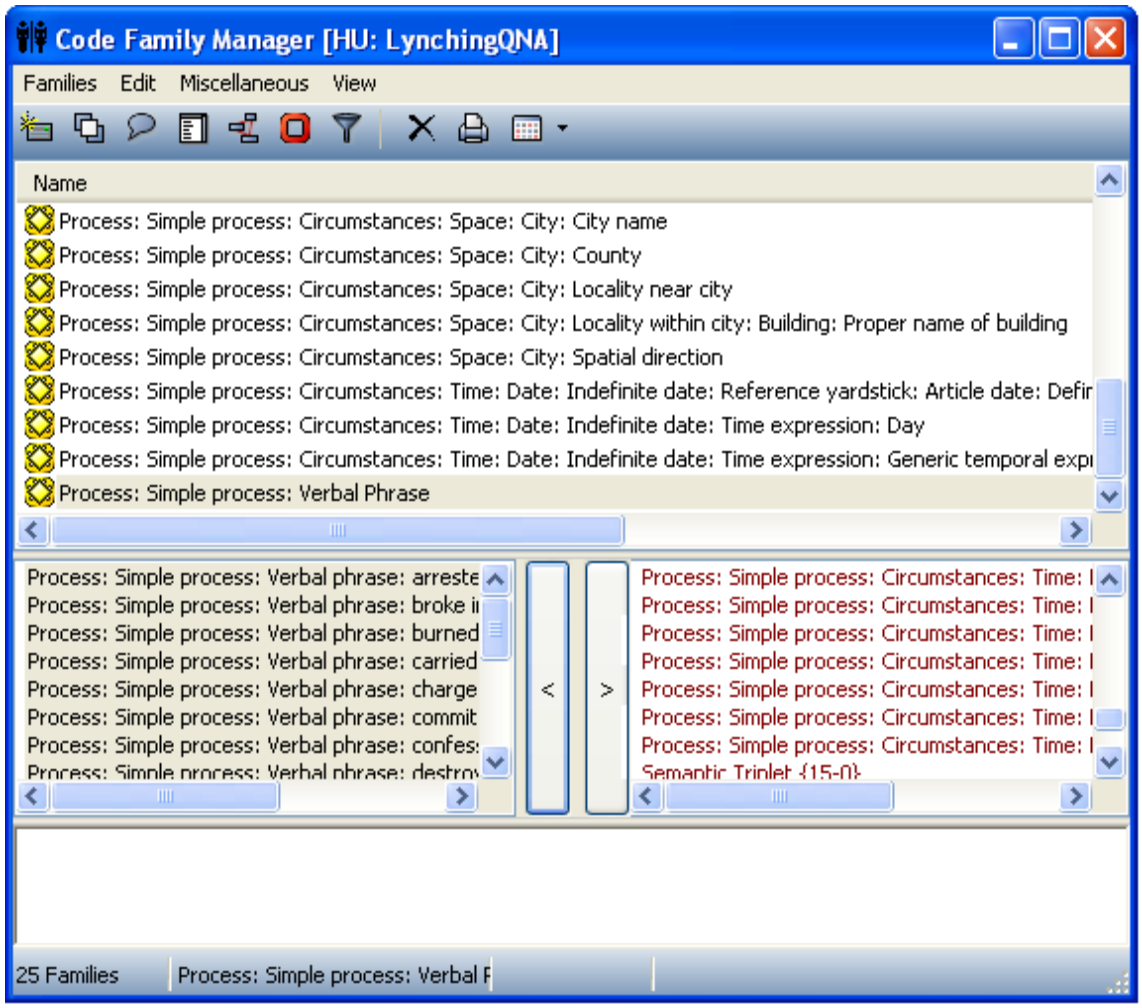

Figure 10: Assigning Codes to a Code Family in ATLAS.ti

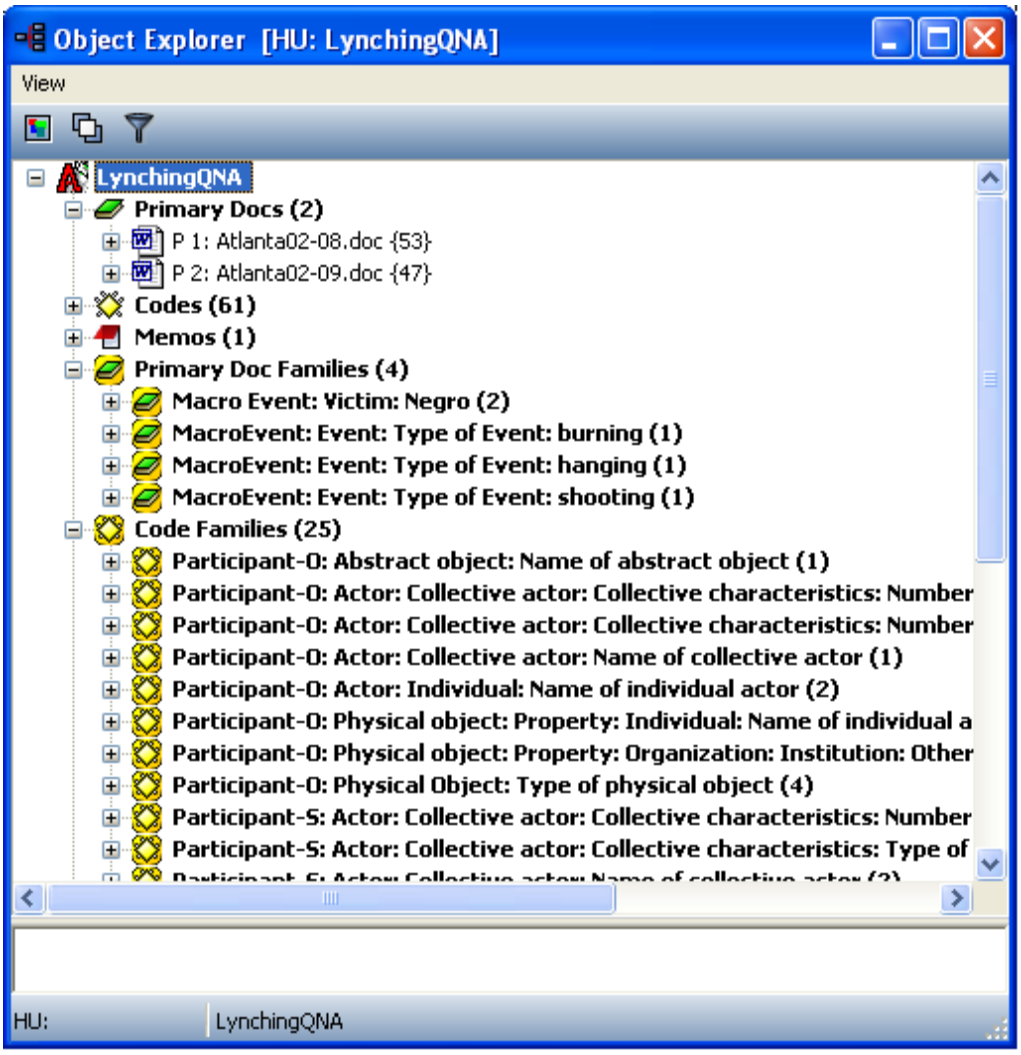

Figure 11: View of the Object Hierarchy in the ATLAS.ti Object Explorer

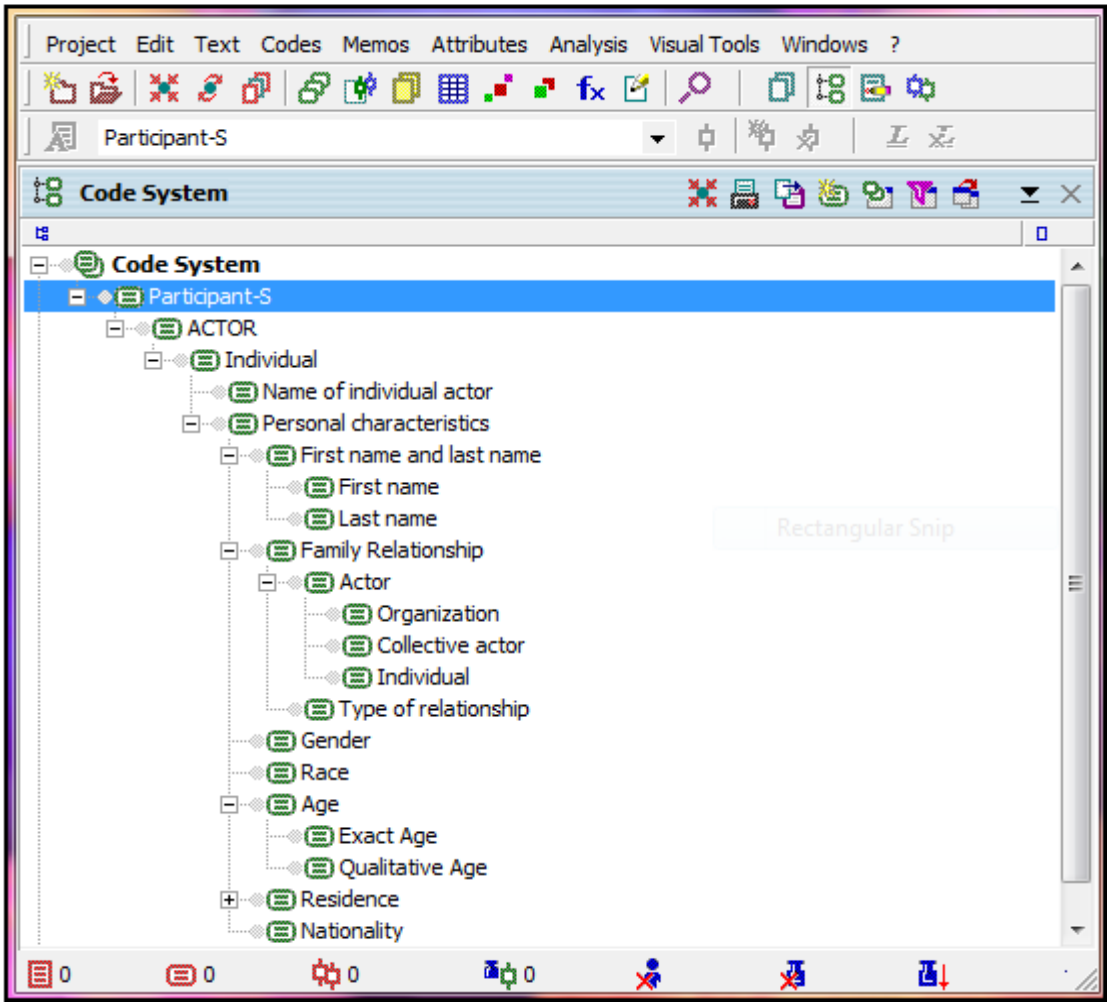

Figure 12: Hierarchical Organization of Categories in MAXQDA

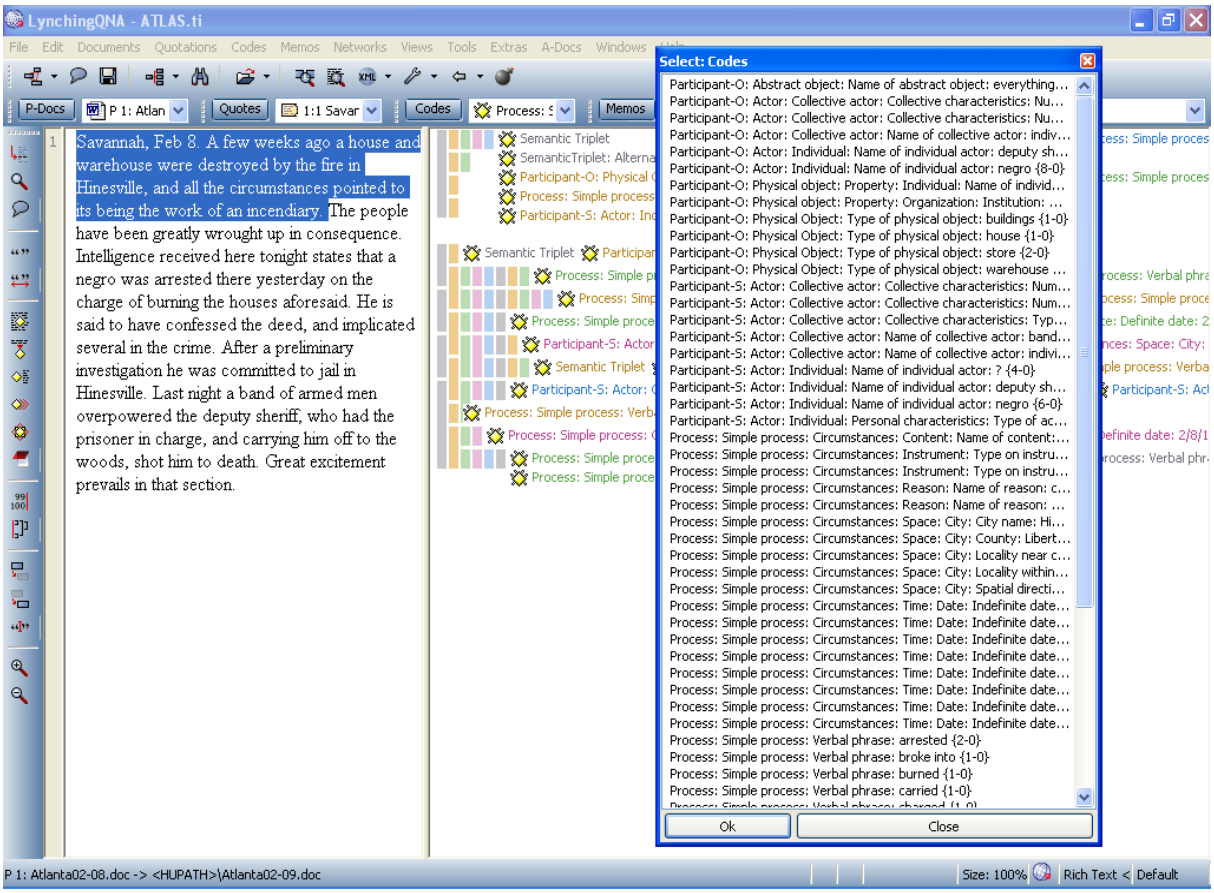

Figure 13: Coding in ATLAS.ti

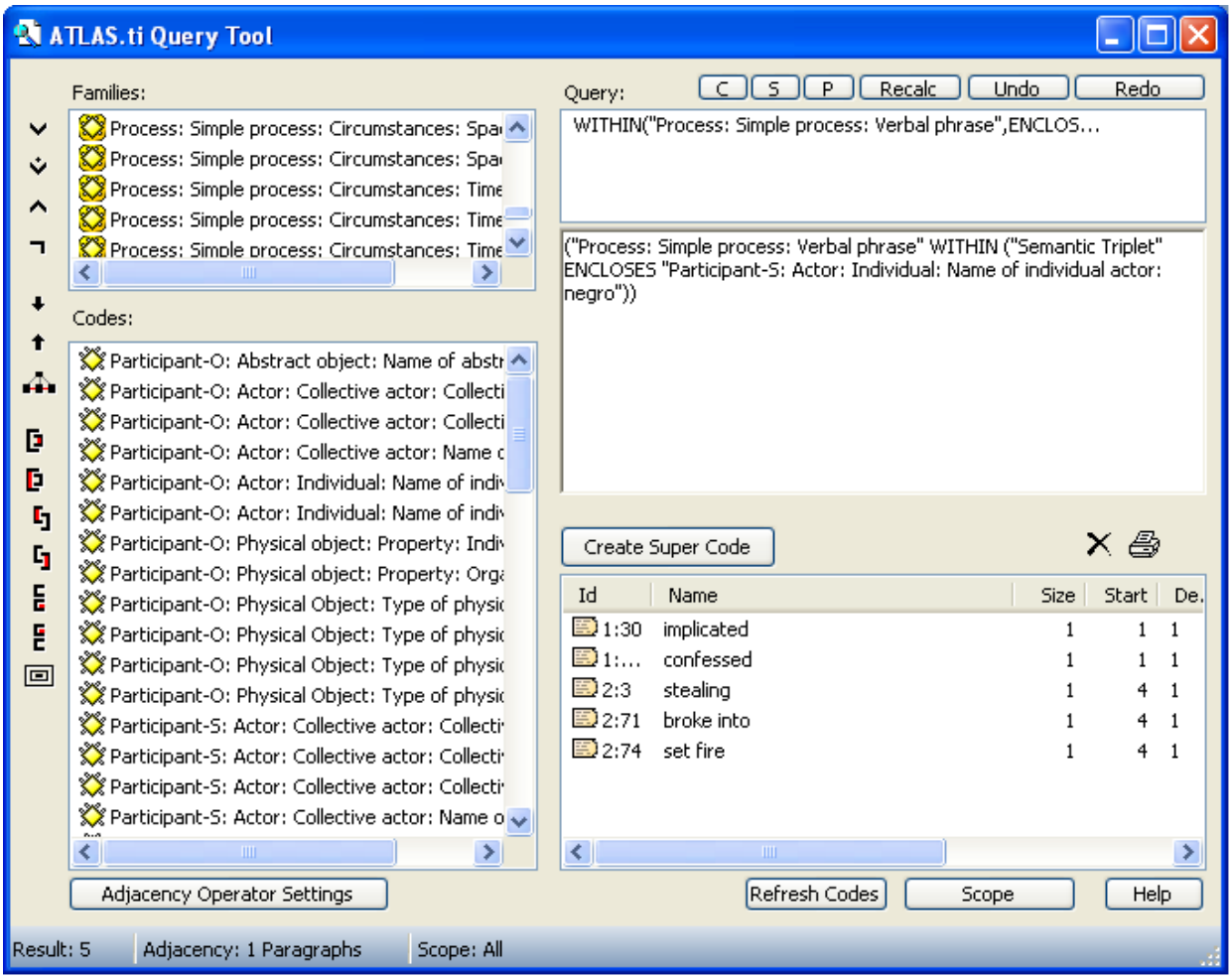

Figure 14: ATLAS.ti Query Tool

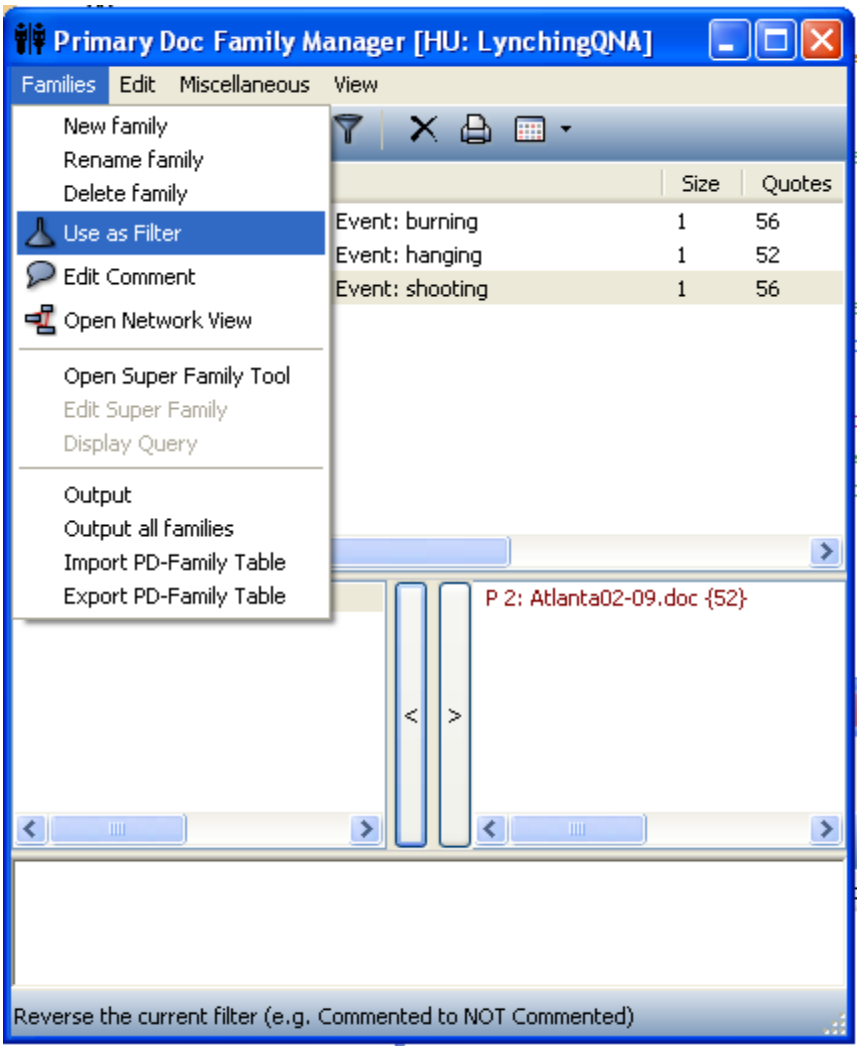

Figure 15: Using a Document Family as a filter in ATLAS.ti

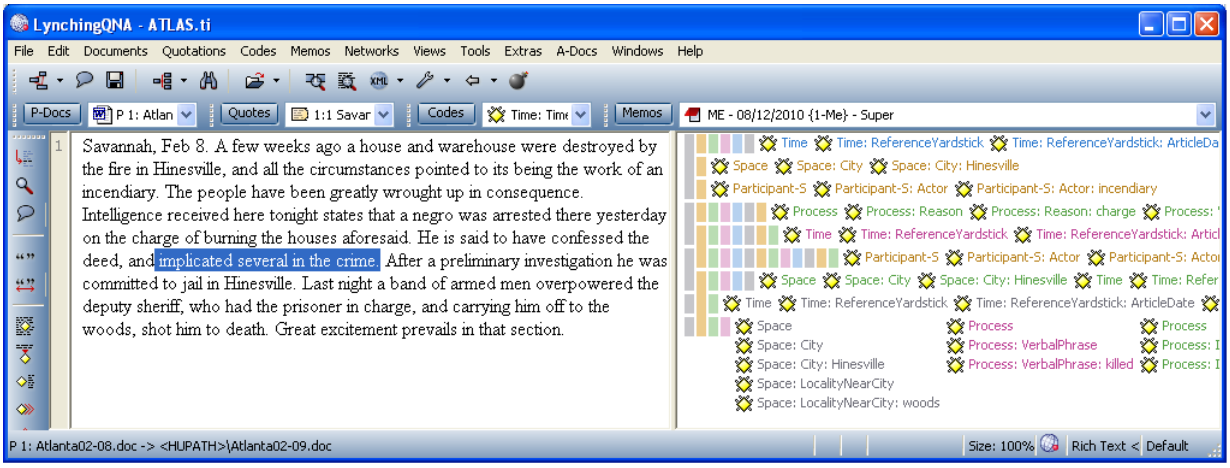

Figure 16: Implicit Elements

Endnotes

[1] Discourse on story grammars has a long tradition, starting from Propp's seminal work on Russian folktales; for a review, see Franzosi (2004: 43-51, 2010: 11-23; in press).

[2] The angular brackets <> denote elements that can be further rewritten; while "terminal elements," i.e., the words or linguistic expressions found in the text, have no <>. Curly brackets {} denote elements that can occur more than one time; while square brackets [] denote optional elements. Thus, in the clause "victim screams" there is only one participant (the agent), while the clause "mob kills negro" has two participants (the agent, mob, and the goal or patient, negro). As a result, the grammar requires only the first participant; the second is optional.

[3] For a complete grammar, see Appendix II.

[4] For a review of positions on the various functional parts of narrative discourse, see Franzosi (2004: 55-59; 2010: 14, 17-20).

[5] As a text, the combined story is not necessarily the same thing as each of the two distinct articles. The original articles stress different factual elements of the story, ultimately, with different ideological slants. The combined story, however, does contain all the factual narrative elements found in the two articles, and it is these factual elements that QNA is interested in.

[6] Assertive, descriptive, and evaluative clauses have been removed from coding and the language of all clauses has been somewhat reworded (e.g., passive sentences have been turned into active ones, inferring the subject/agent when the passive form is missing the agent); on these issues, see Franzosi (2010: 40).

[7] *The Atlanta Constitution*, with a frequency of 473 articles, provides the bulk of the narratives; the next most frequent entry, *The Macon News*, has 46 articles.

[8] We used version 6 of ATLAS.ti, NVivo 9, and MAXQDA 2007.

[9] For a review of these programs, see Lewins and Silver (2007). Other specialized linguistics software has been used by researchers to analyse textual data including Machinese Syntax Software developed by Connexor (Podolny and Hill-Popper, 2004; http://www.connexor.eu/technology/machinese/machinesesyntax), Carley's AutoMap (see for all, Carley, 1993; http://www.casos.cs.cmu.edu/projects/automap/software.html), KEDS and IDEA (see Franzosi, 2010: 62). We chose not to investigate these software options as we were seeking to review programs that are

commonly used, and commercially accessible to many academic researchers (for a comparison of textual analysis software, see Alexa and Zuell. 2000, Popping, 2000).

[10] The Figure displays the SVO structure expressed in terms of a much more complex story grammar grounded in Halliday's functional grammar (see Franzosi, 2010: 20-23; Halliday, 1994), the kind of grammar used in the Georgia lynching project (see Appendix II).

[11] In PC-ACE, an object's "children" are those elements on the right of the arrow in the set of rewrite rules for that object. Vice versa, the object on the left of the arrow is the parent of the objects on the right.

[12] Before starting the actual coding, the user needs to enter identifying information (i.e. newspaper title, page number, etc.) on the source document from which the data is being coded. This will allow PC-ACE to cross-reference each coded object automatically to the correspondent source document.

[13] Since only new information is added to the dictionary, any textual data appears only once in the dictionary (although references to an item may, of course, be multiple).

[14] SQL queries require knowledge of the table structure and table relations. Furthermore, complex queries require nested, iterative SQL queries. Without a Query Manager based on a GUI, SQL is likely to make data retrieval beyond the technical competence of most social scientists.

[15] PC-ACE also allows advanced users to write their own queries.

[16] Data aggregation during and after data entry do not necessarily exclude each other. In fact, they *should* be carried out jointly, where the results of data aggregation during data entry can be checked after data entry and errors can be automatically updated via the Update Manager through the procedure outlined.

[17] It is technically possible to reverse this order, or add both codes and documents as a project develops.

[18] Once you have assigned a set of files to an ATLAS.ti or MAXQDA project, you should abstain from editing them with a different application (e.g. Microsoft Word). Within these CAQDAS programs, in fact, any external editing will only be reflected after re-assigning the files to the project and losing any coding that occurred in the earlier version of the file. Unlike ATLAS.ti and MAXQDA, NVivo can work with files stored either internally or externally to the project database. NVivo internal documents can no longer be edited with external applications. Files stored externally, instead, can still be edited with other programs, the edits also being reflected from within the NVivo project.

[19] In MAXQDA this is called a textgroup and in NVivo this is called a project.

[20] Auto-coding functions are present in ATLAS.ti, NVivo, and MAXQDA. However, decisions required regarding the placement of codes (e.g., passive forms, anaphora resolution problems) can render auto-coding a liability.

[21] It is possible to alter the code order. Codes can be sorted by groundedness (the code frequency), density (the amount of links a code has to other codes), author (who created the code), or when it was created or last modified. These variations do not assist with QNA, so we maintain the default, alphabetical listings.

[22] The objects are: Actor, Address, Age, Content, Event, Family relationship, Implicit object, Individual, Instrument, Macroevent, Number, Organization, Outcome, Ownership, Reason, Relation to other location, Residence, Semantic Triplet, Simple process, Space, and Time. For instance, Actor occurs under 6 complex objects (i.e., Participant-S, Family relationship, Group composition, Subset (among which), Participant-O, and Implicit object). Now, these are the 21 shared children complex with the highest hierarchical position in the grammar. Obviously, their own children also appear in the rewrite rules of their own parents. For instance, Actor occurs under 6 complex objects and so do all its children (i.e., individual, collective actor, organization) and the children of children.

[23] "Macroevent", "Event" and "Semantic triplet" are not rewritten when they appear as a child of a complex object. "Actor" is only rewritten as "Individual", "Collective actor" and "Organization" when it appears as a child of a complex object.

[24] Consider this. An Individual actor can be both under Participant-S and Participant-O, leading to the two codes Participant-S: Actor: Individual: Name of individual actor and Participant-O: Actor: Individual: Name of individual actor. If we have "Negro" and "sheriff" as both Participant-S and Participant-O throughout a text, this would lead to four different codes: Participant-S: Actor: Individual: Name of individual actor: Negro, Participant-S: Actor: Individual: Name of individual actor: sheriff, Participant-O: Actor- Individual: Name of individual actor: Negro, and Participant-O: Actor: Individual: Name of individual actor: sheriff.

[25] MAXQDA and NVivo both have an equivalent to the ATLAS.ti Code Family. In MAXQDA, "sets" are used to arrange the "codes" and "subcodes". In NVivo, relationship nodes can be used to link codes and subcodes, enabling the formation of distinct groups of codes.

[26] Code families also facilitate CAQDAS query operations, as we will see.

[27] CAQDAS researchers do not necessarily have to work with aggregate codes. They can work with disaggregated codes and group these together into aggregated codes. ATLAS.ti, NVivo, and MAXQDA all have functions allowing a user to manipulate existing codes through merging, renaming, duplicating etc. Distinctive to ATLAS.ti is the "super codes" function. This combines codes that have been linked together during querying. Unfortunately, these codes appear as yet more code names in the code list, ultimately defeating the purpose of working with fewer codes.

[28] The number of codes could be further reduced by setting up each object of the story grammar with its own distinct code name, without path and value embedded in its name. Users would then assign several different codes to the same element in the text. For those cases of grammars where the same object has different paths (to repeat, Actor found both under Participant-S and Participant-O, or City found under both Actor and Process), this approach reduces the number of codes required to implement QNA in CAQDAS. For instance, under the first method, the 213 codes for all Participant-S children objects have to be duplicated, with the leading object Participant-O instead of Participant-S, when setting up the codes. Considering the additional codes for Case and Abstract and Physical objects, Participant-O pathways alone require 239 codes. In the alternative method, the same codes are used for both Participant-S and Participant-O children objects with a drop in the number of codes from 901 to 150. And these totals are only baseline figures that do not take into account the values these codes take on and that require hard coding. In spite of the apparent appeal of this alternative approach, there are several drawbacks. First, the approach will increase coding time and decrease data reliability. In our first-proposed method, only one code (e.g., "Process: Simple process: Verbal Phrase: Stole") is used to code the relevant elements in the text. As the code list is arranged alphabetically in CAQDAS programs, finding and applying this code is easy. In the alternative approach, the multiple codes that need to be assigned to the same textual element will not necessarily be found together in an alphabetically-sorted code list (consider "Process", "Simple process", "Verbal Phrase" and "Stole"). Coding will require scrolling through a long list of codes (some 400 of them without counting either the 1266 distinct verbs or the 60 aggregate ones). The longer the list the more time consuming and error prone coding becomes. Second, querying as well becomes more problematic. Consider the query for "WHAT did the "negro" do?" Under the first method, the query is:

("Process: Simple process: Verbal Phrase" WITHIN ("Semantic Triplet" ENCLOSES "Participant-S: Actor: Individual: Name of individual actor: Negro"))

This simple query becomes more complex when using this alternative method. Under the alternative coding method, the query is:

(((("Process" & "Verbal Phrase") & "Simple process") WITHIN ("Semantic Triplet" ENCLOSES ((((("Participant-S" & "Actor") & "Name of individual actor") & "Negro"))

In the first method, 3 codes and 2 operators are involved in obtaining the answer (codes: "Process: Verbal Phrase", "Semantic Triplet", "Participant-S: Actor: Negro"; operators: WITHIN, ENCLOSES). In the second, 9 codes and 8 operators are involved (codes: "Process", "Verbal Phrase", "Simple process", "Semantic Triplet", "Participant-S", "Actor", "Individual", "Name of individual actor", "Negro"; operators: &, which stands for AND, WITHIN, ENCLOSES, &, &, &, &). Query time grows exponentially with query complexity (as measured by the number of codes and operators in the query).

[29] Similarly, in MAXQDA, documents can be organized by text groups or text sets. Attributes can then be applied to texts within these. In NVivo, sources need to be coded as cases. Then the case can be given attribute values.

[30] This tool mirrors the Code Family Manager, as described previously, but it groups documents rather than codes.

[31] CAQDAS programs use the term "hierarchy" and "hierarchical object" slightly differently from PC-ACE, and the way we have used the term up to here. In CAQDAS programs, any complex object is a "hierarchical object." In PC-ACE, only a handful of complex objects are "hierarchical objects" (namely, macroevent, event, semantic triplet). Technically, using PC-ACE's RDBMS terminology, a complex object is characterized by one-to-few relationships; hierarchical objects by one-to-many relationships. In practice, there is little difference between the two types of objects and the distinction only serves design purposes (e.g., in PC-ACE, one-to-many objects are displayed in a different form).

[32] ATLAS.ti makes no distinction between the segments of text which were assigned to our "Participant-S" and our "Participant-O". A query based upon a code "negro" would retrieve all quotations for "negro", whether they were intended as "Participant-S" or "Participant-O". This means that when coding using a QNA approach in ATLAS.ti the codes have to become 1) "Participant-S: Actor: Individual: Name of individual actor: Negro" and "Participant-O: Actor: Individual: Name of individual actor: Negro" or 2) either "Participant-S" or "Participant-O" and all the

codes; “Actor”, “Individual”, “Name of individual actor”, “Negro”. In essence, we build the hierarchy into the code or build it using the query tool.

[33] In MAXQDA this ordering is reversed. The text appears on the right. The codes that have been applied to the text appear on the left. In NVivo the way the panes are displayed can be altered. The default layout is the same as for ATLAS.ti (text on the left, coding on the right). However, altering the layout to that found in MAXQDA (text to the right, coding to the left) will permit more data to display in the document pane.

[34] This is only necessary the first time the code is used; thereafter, it is processed automatically by the code family tool.

[35] The results from this query show that, for the purpose of QNA, it is necessary to code small chunks of text, closely overlapping with grammar simplex objects. If larger units of analysis had been used, such as entire semantic triplets, the query tool would not produce a list of the actions (verbal phrases) associated with an actor. Each result would consist of an entire semantic triplet that would require checking (manually) for the desired objects. Thus, although each CAQDAS program has the capability for selecting much larger sections of text, such an approach is not advisable for QNA.

[36] For example, a query would change from: “Process: Simple Process: Verbal phrase” from “WITHIN” semantic triplets “ENCLOSING” the code “Participant-S: Actor: Individual: Name of individual actor: Negro” to the more complex query: “Process: Simple Process: Verbal phrase” from “WITHIN” semantic triplets “&” “Process: Simple process: Circumstances: Time: Date: Indefinite date: Reference yardstick: Newspaper date” “ENCLOSING” the code “Participant-S: Actor: Individual: Name of individual actor: Negro”.

[37] Hyperlinking in ATLAS.ti does not provide a complex enough set of relationships to enable QNA. However, it does have a limited ability for producing output from hyperlinks. Generally, it only works for simple retrievals but there is a “trick”. Using the query tool it is possible to generate a query and then save it as a super code. Subsequently, hyperlinks can be retrieved from that super code

[38] If 200,000 semantic triplets are a daunting number for any dissertation, 7,000 is well within reach. Doyle (Doyle) coded 8,483 semantic triplets from a sample of best-selling children books; Vicari (2008) coded 5,462 from a sample of social fora; De Fazio (2011) coded 6,035 semantic triplets from published chronology, and Junker’s (2012) use of Chinese activists’ publications and Internet postings has generated some based on some 1,500 triplets.

All of these dissertations brought out novel patterns in the data on the basis of a few thousand triplets.

[39] At the 1st International Seminar on Computer-Aided Qualitative Research, held in Amsterdam, 10-11 June 2008, Stewart Shuman, director of the Qualitative Data Analysis Program at the University of Massachusetts, told the audience of his ATLAS.ti Masterclass that he would not use more than 10 codes and that if a client wanted more than 10 codes he would go through the text twice with different codes rather than increase the code list. Suzanne Friese, of ATLAS.ti, states: "Some researchers develop 40 codes, others a few hundred or even a few thousand codes. … The software … just offers functions to create new codes, to delete, to rename or to merge them. … Developing a lot of codes is clearly an adverse effect of using software. No one would ever come close to 1000 or more codes when using the old-style paper & pencil technique. But also when using software, too many codes lead to a dead end. There might be exceptions, but in most cases this hinders further analysis." (Friese, 2011: 12)

[40] Relations between coding categories (or codes) are central not only in QNA story grammars, but also in other types of approaches to text (e.g. conceptual relationships; on these issues, see Carley, 1993, Franzosi, 2010: 51). These types of complex coding schemes require both the generation of relevant coding categories and the mapping of hierarchical and relational ties between those categories. Since CAQDAS coding schemes can only be generated as lists of codes, users cannot develop complex nested codes in these programs. As a result, CAQDAS programs are not only unsuitable for QNA but also for other forms of textual analyses, such as map analysis, where the relations among different coding categories are of central interest.