Ontologically Parsing Semi-structured Text with Unknown Grammars

David W. Embley[1,2], Stephen W. Liddle[1], Deryle W. Lonsdale[1], Gary J. Norris[2] and Scott N. Woodfield[1,2]

[1]*Brigham Young University, Provo, Utah, USA*
[2]*FamilySearch International, Lehi, Utah, USA*

Abstract
Semi-structured text often contains valuable information that could be parsed and placed in a structured store that is relatively easy to process, but parsing requires a grammar, which is usually unknown. This makes potentially rich information sources such as family history books difficult to access and query, especially for the task of learning about specific individuals of interest. A conceptual modeling approach is useful because we can conceptualize the desired individual records as an ontology relating lexical objects to one another, lexicalize the text, generate a grammar to parse the stream of lexical objects, and compile records that satisfy the grammar and populate the ontological conceptualization. The induced grammars are regular and thus parse in linear time, and we have measured high accuracy (f-score > 90%) for a variety of family history books with minimal effort and expertise required to generate thousands of records per book.

Keywords
ontological conceptualization, grammar, semi-structured text

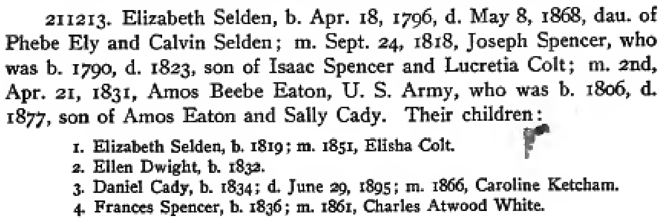
1. Introduction

Aiding discovery of our ancestral past is a booming business, with successful companies such as Ancestry, FamilySearch, Find My Past, MyHeritage, and many more. Beyond heightened interest in discovering our roots, genealogies are also important in the study of inherited genetic disorders, inter-generational poverty, and community and society longevity research [1].

Among the many billions of primary and secondary genealogical sources are record collections compiled by genealogical enthusiasts documenting inter-generational families in ancestral lines, communities, and mortuary or cemetery records. These record collections are written in a semi-structured style amenable to automated information extraction.

Fig. 1 shows a semi-structured text snippet taken from *The Ely Ancestry* [2]. Although not in a formal tabular structure, there are markers such as "b.", "d.", and "dau. of" that identify attribute-value pairs for record fields and a layout that allows these fields to be grouped into records. It thus satisfies the informal definition of semi-structured text that enables identification of fully structured records like the following:

ER2022 Forum, October 17–20, 2022, Hyderabad, India
embley@cs.byu.edu (D. W. Embley); liddle@byu.edu (S. W. Liddle); lonz@byu.edu (D. W. Lonsdale); woodfiel@cs.byu.edu (S. N. Woodfield)


211213. Elizabeth Selden, b. Apr. 18, 1796, d. May 8, 1868, dau. of Phebe Ely and Calvin Selden; m. Sept. 24, 1818, Joseph Spencer, who was b. 1790, d. 1823, son of Isaac Spencer and Lucretia Colt; m. 2nd, Apr. 21, 1831, Amos Beebe Eaton, U. S. Army, who was b. 1806, d. 1877, son of Amos Eaton and Sally Cady. Their children:

1. Elizabeth Selden, b. 1819; m. 1851, Elisha Colt.
2. Ellen Dwight, b. 1832.
3. Daniel Cady, b. 1834; d. June 29, 1895; m. 1866, Caroline Ketcham.
4. Frances Spencer, b. 1836; m. 1861, Charles Atwood White.

Figure 1: Text Snippet from *The Ely Ancestry* [2].

Person((Name, “Elizabeth Selden”), (BirthDate, “Apr. 18, 1796”), (DeathDate, “May 8, 1868”))

Person((Name, “Joseph Spencer”), (BirthDate, “1790”), (DeathDate, “1823”))

Marriage((Name, “Frances Spencer”), (MarriageDate, “1861”), (Spouse, “Charles Atwood White”))

Our approach to automatically extracting records from semi-structured books is to rewrite the book in terms of the types of records we seek. Record fields are attribute-value pairs, which are grouped according to the ontologically conceptualized record types we seek. The properties of conceptualized relationship sets allow us to generate grammars for each record type, which we can then use to check, parse, and compile records.

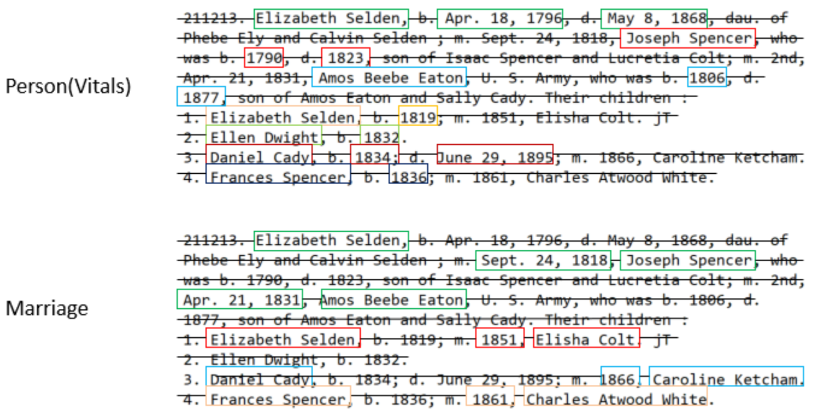
Fig. 2 shows a rewriting of the text snippet in Fig. 1 with respect to two different record types. The lined-through text is suppressed leaving an ordered list of record field values, which when labeled become attribute-value pairs. Observe: (1) the group of value fields that belong to a record are partitioned into record groups in the linear flow of the text, and (2) every field value beyond the first has a direct relationship to the first field value and can be seen as a property of the entity denoted by the first field value. Record examples in these rewritings include:

In *Person*: “Elizabeth Selden” is the first field value, “Apr. 18, 1796” is her birth date, “May 8, 1868” is her death date.

In *Marriage*: “Elizabeth Selden” was married on “Sept. 24, 1818” to “Joseph Spencer” and was married on “Apr. 21, 1831” to “Amos Beebe Eaton”.

Valid sentences with respect to a generated grammar for a record type can be parsed and compiled into relational database records. Tables 1 and 2 show the results for the text snippet in Figs. 1 and 2 respectively.

Tables 1 and 2 show coordinates for some text values, which serve as unique identifiers. For example, the Elizabeth Selden born in 1796 has coordinates “(@563,33,4)” and she is distinct from the Elizabeth Selden born in 1819 who has coordinates “(@563,38,4)”. Coordinates are built from page, line, and token offset values for the corresponding lexical tokens. *Name* is a key for the *Person* table, and *Name-Spouse* is a compound key for the *Marriage* table. Unique coordinates also allow us to join records appropriately, thus deriving more complete, linked information about individuals, marriages, and children.



~~211213.~~ Elizabeth Selden, ~~b.~~ Apr. 18, 1796~~, d.~~ May 8, 1868~~, dau. of~~
~~Phebe Ely and Calvin Selden ; m. Sept. 24, 1818,~~ Joseph Spencer~~, who~~
~~was b.~~ 1790~~, d.~~ 1823~~, son of Isaac Spencer and Lucretia Colt; m. 2nd,~~
~~Apr. 21, 1831,~~ Amos Beebe Eaton~~, U. S. Army, who was b.~~ 1806~~, d.~~
1877~~, son of Amos Eaton and Sally Cady. Their children :~~
~~1.~~ Elizabeth Selden~~, b.~~ 1819~~; m. 1851, Elisha Colt. jT~~
~~2.~~ Ellen Dwight~~, b.~~ 1832.
~~3.~~ Daniel Cady~~, b.~~ 1834~~; d.~~ June 29, 1895~~; m. 1866, Caroline Ketcham.~~
~~4.~~ Frances Spencer~~, b.~~ 1836~~; m. 1861, Charles Atwood White.~~

~~211213.~~ Elizabeth Selden, ~~b. Apr. 18, 1796, d. May 8, 1868, dau. of~~
~~Phebe Ely and Calvin Selden ; m.~~ Sept. 24, 1818~~,~~ Joseph Spencer~~, who~~
~~was b. 1790, d. 1823, son of Isaac Spencer and Lucretia Colt; m. 2nd,~~
Apr. 21, 1831~~,~~ Amos Beebe Eaton~~, U. S. Army, who was b. 1806, d.~~
~~1877, son of Amos Eaton and Sally Cady. Their children :~~
~~1.~~ Elizabeth Selden~~, b. 1819; m.~~ 1851~~,~~ Elisha Colt~~. jT~~
~~2. Ellen Dwight, b. 1832.~~
3. Daniel Cady~~, b. 1834; d. June 29, 1895; m.~~ 1866~~,~~ Caroline Ketcham.
~~4.~~ Frances Spencer~~, b. 1836; m.~~ 1861~~,~~ Charles Atwood White.

Figure 2: Record Types Partition Running Text.

Table 1
Person Relational Database Table.

Person(	Name	BirthDate	DeathDate	)
	Elizabeth Selden (@563,33,4)	Apr. 18, 1796	May 8, 1868	
	Joseph Spencer (@563,34,16)	1790	1823	
	Amos Beebe Eaton (@563,36,6)	1806	1877	
	Elizabeth Selden (@563,38,4)	1819	-	
	Ellen Dwight (@563,39,4)	1843	-	
	Daniel Cady (@563,40,4)	1834	June 29, 1895	
	Frances Spencer (@563,41,4)	1836	-	

Table 2
Marriage Relational Database Table.

Marriage(	Name	Spouse	MarriageDate	)
	Elizabeth Selden (@563,33,4)	Joseph Spencer (@563,34,16)	Sept. 24, 1818	
	Elizabeth Selden (@563,33,4)	Amos Beebe Eaton (@563,36,6)	Apr. 21, 1831	
	Elizabeth Selden (@563,38,4)	Elisha Colt (@563,38,15)	1851	
	Daniel Cady (@563,40,4)	Caroline Ketcham (@563,40,22)	1866	
	Frances Spencer (@563,41,4)	Charles Atwood White (@563,41,15)	1861	

In this paper we present the following contributions, which are centered on a conceptual modeling approach. In Section 2 we describe a theoretical foundation for compiling database records by parsing semi-structured text whose grammar is unknown but can be derived from an ontological conceptualization of the information conveyed in the text. In Section 3 we present theoretical extensions for parsing semi-structured text with nested records and anomalous field values. In Section 4 we describe an implementation of the theory and its extensions that achieves high extraction accuracy for both named entity recognition (NER) and named relation recognition (NRR) with relatively modest required effort and expertise.

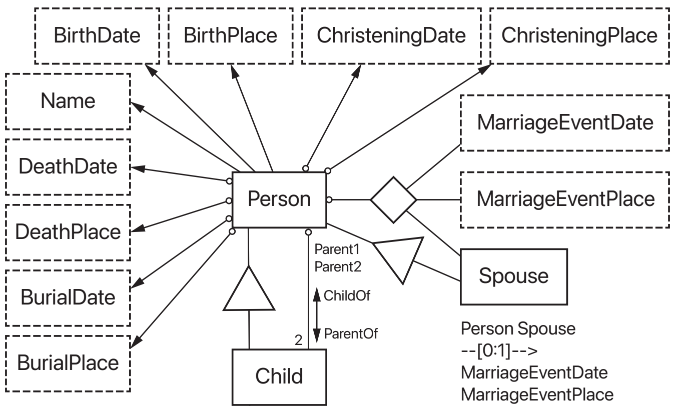


Figure 3: Hypergraph Diagram of an OSM Extraction Ontology [3]

Table 3
Nested ParentOf Relational Database Table.

ParentOf(	Name	(Child)*	)
	Elizabeth Selden (@563,33,4)	Elizabeth Selden (@563,38,4)	
		Ellen Dwight	
		Daniel Cady	
		Frances Spencer	

2. Theoretical Foundations

Conceptual modeling is at the core of this work: our approach starts with an ontology that corresponds precisely to a real-world application, and thus it constrains the text of semi-structured documents written for this application to correspond to a small number of records types. Book authors who wish to write in a semi-structured style must necessarily write with respect to these record types. Fig. 3 shows a diagram of the family history ontology (a conceptual model instance) that we use in our running example. Authors of family history books who wish to describe genealogies must necessarily write information about parents, children, dates of important life events, etc., which the ontology describes.

The next step of our approach is to render the ontology's hypergraph diagram in Nested Normal Form (NNF) [4]. This entails the set of record types for a book and establishes the basis for generating grammars for each of these record types. A relational database schema is a set of relation schemes, each of which is a named set of attribute names. Relation schemes are normalized, which eliminates redundancy with respect to applicable functional dependencies among its attributes while maximizing the number of attributes in each relation scheme. Although much less common in practice, relation schemes can have groups of one or more of its attributes designated as being nested with respect to the non-nested attributes. If, for example, *Child* were to be nested in a *ParentOf* table, it could be rendered as Table 3. Nested relations also have an NNF normal form, which leads us directly to the grammars we wish to derive for parsing semi-structured text.

The OSM hypergraph diagram in Fig. 3 is reduced with respect to its embedded functional dependencies [5]. All relationship sets are binary except for the 4-ary relationship set, which cannot be losslessly reduced to binary relationship sets because of the compound left-hand side of the functional dependency, *Person Spouse* → *MarriageEventDate MarriageEventPlace*. As such, the derivation of an NNF relational database schema is immediate. NNF relation schemes generated from the ontology include the following:

Person(Name, BirthDate, BirthPlace, ChristeningDate, ChristeningPlace, DeathDate, DeathPlace, BurialDate, BurialPlace)

Marriage(Name, (Spouse, MarriageDate, MarriagePlace)*)

ParentOf(Parent, (Child)*)

Family(Parent1, Parent2, (Child)*)

ChildOf(Child, Parent1, Parent2)

Note that *Family* subsumes *ParentOf* where *Parent2* is always empty. Formally, we use one or the other but not both for a book, and which we use depends on the book's layout.

Given the NNF record types, grammar generation is straightforward. We instantiate objects in a non-lexical object set *N* (cause them to come into existence in *N*) by ontological commitment of a text phrase in one or more lexical object sets connected by relationship sets to *N*. For the ontology diagram in Fig. 3, objects are instantiated in the non-lexical object set *Person* when text snippets are added to the connected lexical object set *Name*. Moreover, since *Child* and *Spouse* are specializations of *Person*, their ontological commitment is also by *Name*. Specializations are populated only by explicitly designating one of the objects in the generalization *Person* to be an element of a specialization.

The grammar immediately falls out from these scheme trees. As an example consider the *Person* records in Fig. 1 where a person has birth and death dates. A person can have either, both, or neither dates, and the dates can come in any order, as follows:

Record Scheme: Person(Name, BirthDate, DeathDate)

Grammar: <Person> → Name | Name BirthDate | Name DeathDate | Name BirthDate DeathDate | Name DeathDate BirthDate

Grammar: <Person> → Name (ϵ | BirthDate | DeathDate | BirthDate DeathDate | DeathDate BirthDate)

Grammar: <Person> → Name [BirthDate, DeathDate]

The three grammar production rules are equivalent such that the bracket notation denotes an alternation of the power set of the set of listed terminals with each subset within the power set ordered in all permutations. Using square bracket notation, we give the grammar for all of our record types:

Record Scheme: Person($\underline{\text{Name}}$, BirthDate, BirthPlace, ChristeningDate, ChristeningPlace, DeathDate, DeathPlace, BurialDate, BurialPlace)

Grammar: <Person> → Name [BirthDate, BirthPlace, ChristeningDate, ChristeningPlace, DeathDate, DeathPlace, BurialDate, BurialPlace]

Record Scheme: Marriage($\underline{\text{Name, (Spouse}}$, MarriageDate, MarriagePlace)*)

Grammar: <Marriage> → Name ($\underline{\text{[Spouse}}$, MarriageDate, MarriagePlace])$^+$

(The underline in this notation adds the requirement that the underlined terminal symbol appear in every attribute group; alternatively, it could be thought of as eliminating from the power set every subset that that does not include the underlined terminal.)

Record Scheme: ParentOf($\underline{\text{Parent, (Child)}^*}$)

Grammar: <ParentOf> → Parent1 (Child)$^+$

Record Scheme: Family($\underline{\text{Parent1, Parent2, (Child)}^*}$)

Grammar: <Family> → Parent1 Parent2 (Child)$^+$

Record Scheme: ChildOf($\underline{\text{Child}}$, Parent1, Parent2)

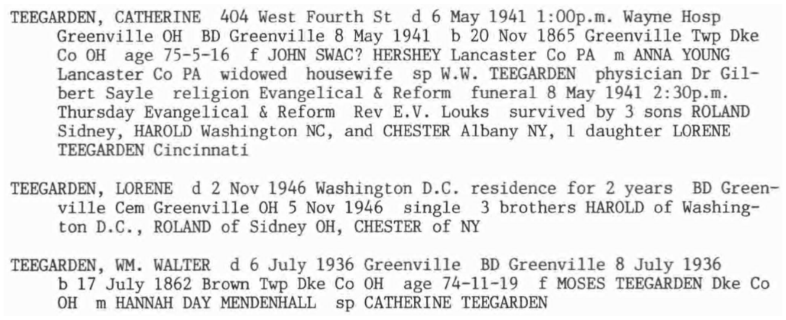
Grammar: <ChildOf> → Child Parent1 Parent2

These grammars are all regular in Chomsky's classification, which thus allows for efficient linear-time processing [6, 7].

The next step is to perform lexical analysis. For each record type *R*, a lexer rewrites the book as a sequence of attribute-value pairs corresponding to the fields of *R*. The values in the attribute-value pairs all have book coordinates (described earlier and illustrated in Table 3). The lexer generates attribute-value pairs such as *Name*: *Elizabeth Selden*, *BirthDate*: *Apr. 18, 1796*, and *DeathDate*: *May 8, 1868*. The lexer is programmed by example. A programmer specifies the attribute (e.g. *Name*) and copies a snippet from the text that contains the value and enough context to uniquely categorize and thus label it. The text snippet is then categorically tokenized. After the entire text of a book has been tokenized, the lexer finds matching patterns and labels the book's running text.

Parsing text conforming to regular grammars is straightforward, as is compiling records. The compiler knows how to relate the values in the attribute-value pairs because every binary relation is formed by relating the value associated with the record-head attribute with the value associated with each of the other attributes, and every n-ary ($n > 2$) relation is formed by relating the value associated with the record-head attribute with the group of $n - 1$ values of the other $n - 1$ attributes in the n-ary relation.

Note that parse errors are valuable in this application. For example, when parsing the *Person* grammar, encountering a second birth date is a parse error as is a second of all the other functionally dependent attributes. These errors help us detect missing record heads. If, for


TEEGARDEN, CATHERINE 404 West Fourth St d 6 May 1941 1:00p.m. Wayne Hosp Greenville OH BD Greenville 8 May 1941 b 20 Nov 1865 Greenville Twp Dke Co OH age 75-5-16 f JOHN SWAC? HERSHEY Lancaster Co PA m ANNA YOUNG Lancaster Co PA widowed housewife sp W.W. TEEGARDEN physician Dr Gilbert Sayle religion Evangelical & Reform funeral 8 May 1941 2:30p.m. Thursday Evangelical & Reform Rev E.V. Louks survived by 3 sons ROLAND Sidney, HAROLD Washington NC, and CHESTER Albany NY, 1 daughter LORENE TEEGARDEN Cincinnati

TEEGARDEN, LORENE d 2 Nov 1946 Washington D.C. residence for 2 years BD Greenville Cem Greenville OH 5 Nov 1946 single 3 brothers HAROLD of Washington D.C., ROLAND of Sidney OH, CHESTER of NY

TEEGARDEN, WM. WALTER d 6 July 1936 Greenville BD Greenville 8 July 1936 b 17 July 1862 Brown Twp Dke Co OH age 74-11-19 f MOSES TEEGARDEN Dke Co OH m HANNAH DAY MENDENHALL sp CATHERINE TEEGARDEN

Figure 4: Three *Miller Funeral Home Records* [8].

example, we are parsing *Person* records in Fig. 2 and find that child Ellen Dwight has birth dates of 1832 *and* 1834, we know that Daniel Cady's name has not been found and labeled. Another kind of parse error can occur in the *Marriage* grammar. Note that in the *Marriage* grammar the star in the record scheme becomes a plus, and thus it is a syntax error if there is no *Spouse*. Detecting these types of errors is valuable in our quest to find and extract name instances for all mentioned persons in a book.

The key idea that makes parsing and compiling straightforward is that all grammars begin with a terminal symbol, which we call the record head (i.e. respectively, *Name*, *Name*, *Parent1*, *Parent1*, *Child* for the five grammars above). Then for a recognized record-head object every other recognized object has a direct relationship to the head (in the case of *Marriage* it is each spouse's *Name-MarriageDate-MarriagePlace* group that relates to the record-head object).

This principle lets us group grammars together so long as it holds. Consider the text snippet in Fig. 4, which is a record for persons who have died and who have been taken care of by a funeral home. Observe that for each person *p* all information in *p*'s record has a direct relationship to *p*. In this case we can group grammars for *Person*, *Marriage*, *ParentOf*, and *ChildOf* as one grammar:

Record Scheme: Individual(Name, BirthDate, BirthPlace, ChristeningDate, ChristeningPlace, DeathDate, DeathPlace, BurialDate, BurialPlace, (Spouse, MarriageDate, MarriagePlace)*, (Child)*, Parent1, Parent2)

Grammar: <Individual> → Name [BirthDate, BirthPlace, ChristeningDate, ChristeningPlace, DeathDate, DeathPlace, BurialDate, BurialPlace, ([Spouse, MarriageDate, MarriagePlace])*, (Child)*, Parent1, Parent2]

From Fig. 4 we can extract the following attribute-value pairs for Catherine Teegarden by parsing the text according to the grammar above:

Individual((Name, [TEEGARDEN, CATHERINE (@343,20,2)]), (DeathDate, [6 May 1941]), (DeathPlace, [Greenville OH]), (BurialPlace, [Greenville]), (BurialDate, [8 May 1941]), (BirthDate, [20 Nov 1865]), (BirthPlace, [Greenville Twp Dke Co OH]), (Parent1, [JOHN SWAC? HERSHEY]), (Parent2, [ANNA YOUNG]), ((Spouse, [W.W. TEEGARDEN]), (MarriageDate, []), (MarriagePlace, [])), ((Child, [ROLAND]), (Child, [HAROLD]), (Child, [CHESTER]), (Child, [LORENE TEEGARDEN])))

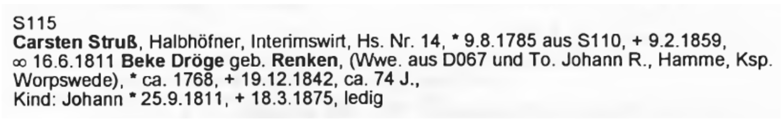

S115
Carsten Struß, Halbhöfner, Interimswirt, Hs. Nr. 14, * 9.8.1785 aus S110, + 9.2.1859,
∞ 16.6.1811 **Beke Dröge** geb. **Renken**, (Wwe. aus D067 und To. Johann R., Hamme, Ksp. Worpswede), * ca. 1768, + 19.12.1842, ca. 74 J.,
Kind: Johann * 25.9.1811, + 18.3.1875, ledig

Figure 5: Text snippet from *Flögeln* [10] with *Wwe.* reference.

As with most theoretical formulations, real-world practicalities (in our case mostly author nuances) can inject irregularities which must be considered. For this approach to succeed the text must be *semi-structured* and have two properties: (1) sufficient context to unambiguously form attribute-value pairs for every sequence of tokens of interest, and (2) for each record type *R*, there exists an *R*-record partitioning of the book's text rewritten as a sequence of attribute-value pairs applicable to *R*.

A number of irregularities often make this process interesting. For example, in the Kilbarchan book [9], twins James and William share the same birth date and are listed as "James and William, 9 April 1654". This construction violates our semi-structured record partitioning principle. However, the twins James and William are both on Page 25, Line 53, and there is only one token, namely "and", between them. Our record compiler for *Person* can and does check for twins, and even triplets and quadruplets with shared birth information by finding consecutive names on the same page and line with only one or two tokens between them and birth or christening information following only the last.

3. Theoretical Extensions

In this section we show that we can make the theory more generally applicable, (1) by relaxing the semi-structured requirements and (2) by extending our coverage to include all conceptual-modeling features in [3]. We only need to add aggregation and recursive relationship sets to obtain full coverage of all OSM data modeling constructs described in [3]. Thus the theory extends to record extraction from semi-structured text for any application modeled by OSM. New grammars are introduced, but they remain regular, Chomsky type 3 grammars and thus execution remains linear.

3.1. Labeling Issues

An example of a "labeling issue" occurs in Fig. 5 where "geboren" (abbreviated "geb.") indicates that Beke Dröge's birth surname is Renken. We can capture this information by representing *Name* using an aggregation as Fig. 6 shows. This aggregation is for what we call canonical names—a name with categorized sub-parts. Canonical names are parsed as a record type of their own according to the following grammar:

<CanonicalName> → NamePrefix [Title, FirstName, BirthSurname, MarriedSurname, Suffix]

NamePrefix is the label for the token immediately preceding the name (whatever it is). NamePrefix serves as the record head and is used to associate the canonical name specification with the name to which it applies.

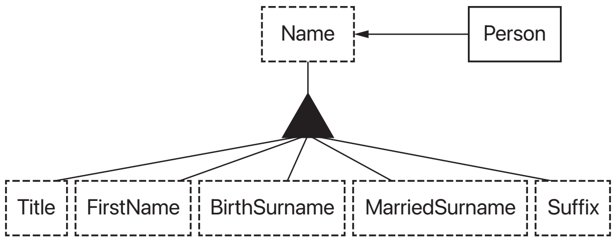
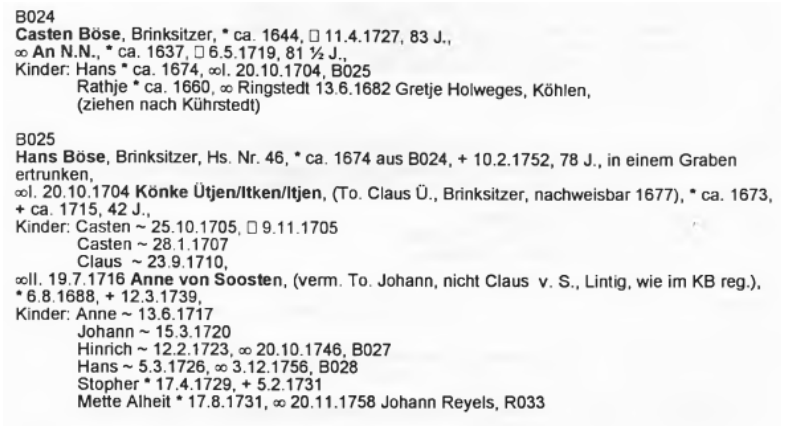


Figure 6: Name Aggregate Extension for the Conceptual Model Diagram in Fig. 3.



B024
Casten Böse, Brinksitzer, * ca. 1644, □ 11.4.1727, 83 J.,
∞ **An N.N.**, * ca. 1637, □ 6.5.1719, 81 ½ J.,
Kinder: Hans * ca. 1674, ∞I. 20.10.1704, B025
Rathje * ca. 1660, ∞ Ringstedt 13.6.1682 Gretje Holweges, Köhlen,
(ziehen nach Kührstedt)

B025
Hans Böse, Brinksitzer, Hs. Nr. 46, * ca. 1674 aus B024, + 10.2.1752, 78 J., in einem Graben ertrunken,
∞I. 20.10.1704 **Könke Ütjen/Itken/Itjen**, (To. Claus Ü., Brinksitzer, nachweisbar 1677), * ca. 1673, + ca. 1715, 42 J.,
Kinder: Casten ~ 25.10.1705, □ 9.11.1705
Casten ~ 28.1.1707
Claus ~ 23.9.1710,
∞II. 19.7.1716 **Anne von Soosten**, (verm. To. Johann, nicht Claus v. S., Lintig, wie im KB reg.), * 6.8.1688, + 12.3.1739,
Kinder: Anne ~ 13.6.1717
Johann ~ 15.3.1720
Hinrich ~ 12.2.1723, ∞ 20.10.1746, B027
Hans ~ 5.3.1726, ∞ 3.12.1756, B028
Stopher * 17.4.1729, + 5.2.1731
Mette Alheit * 17.8.1731, ∞ 20.11.1758 Johann Reyels, R033

Figure 7: Text snippet from *Familienbuch des Kirchspiels Flögeln* [10]—a record book of families in the Flögeln Parish in Lower Saxony, Germany (~1670–1900).

3.2. Record Anomalies

"Anomalies" sometimes occur within the textual space of a record, including reference identifiers, parenthetical remarks, and context switches in which the document's text provides information within the textual space of a record about some other person beside the record head.

The Flögeln snippet in Fig. 7 shows that each household has an identifier that the author uses to refer to related persons. For example, Hans Böse is declared to be "aus B024"—thus asserting that Hans is a member of the household B024. Indeed, he is the same person as the first child in household B024 and hence the reader knows that he is the son of Casten and An Böse. A reverse reference also appears, as Hans in household B024 has a reference to B025, a marriage to Könke Ütjen (or Itken or Itjen). Indeed, his first marriage and the matching marriage date, 20.10.1704, assure us that his spouse has been correctly identified.

The lexer labels these in-line identifiers according to the role they play: B025 is Hans's wife and is labeled as *Spouse*, and B024 is Hans's parent (meaning the head of the household in B024) and is labeled as *Parent1*. Then to process in-line reference identifiers, we seek to replace them by the names of the persons referenced (along, of course, with their book coordinates). This requires that we build an inverted index of reference identifiers to households and then reason about which member of the household is being referenced. Then we replace references with the corresponding names as appropriate.

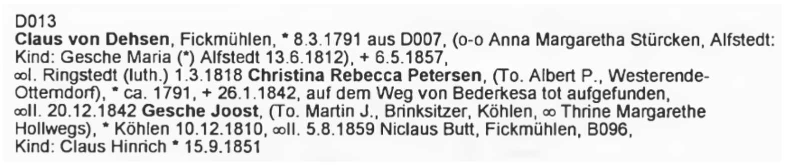

D013
Claus von Dehsen, Fickmühlen, * 8.3.1791 aus D007, (o-o Anna Margaretha Stürcken, Alfstedt:
Kind: Gesche Maria (*) Alfstedt 13.6.1812), + 6.5.1857,
∞I. Ringstedt (luth.) 1.3.1818 **Christina Rebecca Petersen**, (To. Albert P., Westerende-Otterndorf), * ca. 1791, + 26.1.1842, auf dem Weg von Bederkesa tot aufgefunden,
∞II. 20.12.1842 **Gesche Joost**, (To. Martin J., Brinksitzer, Köhlen, ∞ Thrine Margarethe Hollwegs), * Köhlen 10.12.1810, ∞II. 5.8.1859 Niclaus Butt, Fickmühlen, B096,
Kind: Claus Hinrich * 15.9.1851

Figure 8: Text snippet from *Flögeln* [10] with Context Switches.

Flögeln has "context switches" in the sense that a person can be embedded (nested) within another person's text partitioned record space. This nesting constitutes a violation of the semi-structured assumption. Fig. 8 shows an example. Observe that Claus von Dehsen has a spouse, Anna Margaretha Stürcken (as indicated by the o-o symbol, denoting an illegitimate union), and Anna has a child (with Claus, of course, but within Anna's record space) who was born illegitimately (as indicated by the (*) symbol) in the village of Alfstedt on 13.6.1812. The death date 6.5.1857 (indicated by the + symbol) belongs to Claus and thus the record information for both Anna and Gesche is nested textually within Claus's record about his birth and death. It is the parentheses that help us understand the limits of the nesting. Note, however, that the parenthetical remark following Gesche Joost does not contain record information, but rather only her father, Martin J. and her mother Thrine Margarethe Hollwegs as denoted by the symbol "To." (daughter of).

The key to creating a single grammar for semi-structured family documents like Flögeln is to label household members that have two roles in such a way that the parser can recognize both roles. We choose to label them with role names in all-capitalized letters using their role name to designate their relationship to the household head and all-CAP letters to designate that they, themselves, can be record heads. Ontologically, the roles are those non-lexical object sets that denote subsets of *Person*, and thus for our application we have *CHILD*, *SPOUSE*, *PARENT1*, and *PARENT2*. The Flögeln *Person* grammar is:

<Person> → Name [BirthDate, BirthPlace, ChristeningDate, ChristeningPlace, DeathDate, DeathPlace, BurialDate, BurialPlace, LPAREN, RPAREN] <PersonRelation>?

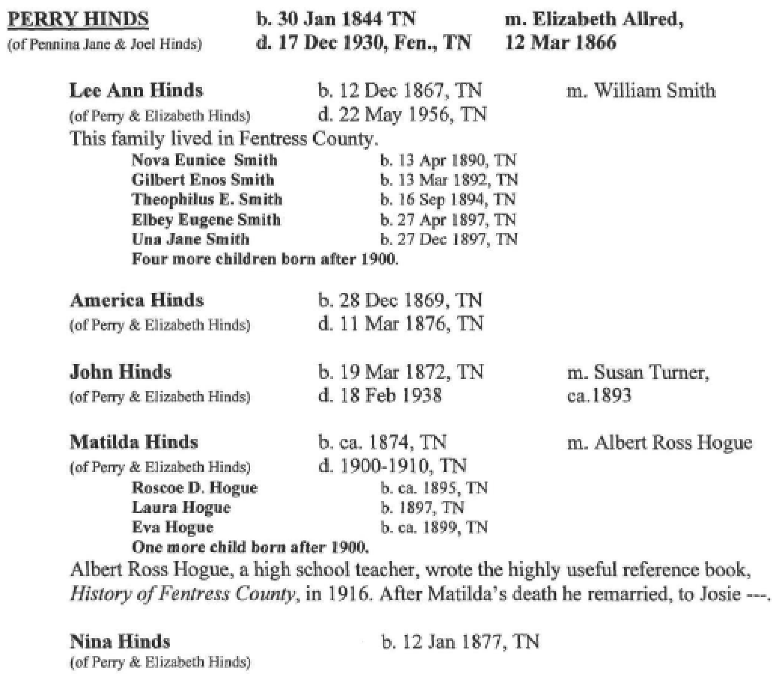
<PersonRelation> → (SPOUSE | CHILD | PARENT1 | PARENT2) [BirthDate, BirthPlace, ChristeningDate, ChristeningPlace, DeathDate, DeathPlace, BurialDate, BurialPlace, LPAREN, RPAREN] <PersonRelation>?

Other grammars for *Marriage*, *ParentOf*, *Family*, and *ChildOf* are extended in similar fashion.

3.3. Recursive Relationships

In some cases, recursive relationships are needed to process textual records that violate the semi-structured properties we assume. For example, consider the Mullinix snippet in Fig. 9, where Lee Ann Hinds is a daughter of Perry and Elizabeth Allred Hinds; Lee Ann's children include Nova Eunice Smith and five others. There is no *a priori* limit on how deeply nested family records could be.

Grammars for Mullinix include the following:


PERRY HINDS b. 30 Jan 1844 TN m. Elizabeth Allred,
(of Pennina Jane & Joel Hinds) d. 17 Dec 1930, Fen., TN 12 Mar 1866

Lee Ann Hinds b. 12 Dec 1867, TN m. William Smith
(of Perry & Elizabeth Hinds) d. 22 May 1956, TN
This family lived in Fentress County.
Nova Eunice Smith b. 13 Apr 1890, TN
Gilbert Enos Smith b. 13 Mar 1892, TN
Theophilus E. Smith b. 16 Sep 1894, TN
Elbey Eugene Smith b. 27 Apr 1897, TN
Una Jane Smith b. 27 Dec 1897, TN
Four more children born after 1900.

America Hinds b. 28 Dec 1869, TN
(of Perry & Elizabeth Hinds) d. 11 Mar 1876, TN

John Hinds b. 19 Mar 1872, TN m. Susan Turner,
(of Perry & Elizabeth Hinds) d. 18 Feb 1938 ca.1893

Matilda Hinds b. ca. 1874, TN m. Albert Ross Hogue
(of Perry & Elizabeth Hinds) d. 1900-1910, TN
Roscoe D. Hogue b. ca. 1895, TN
Laura Hogue b. 1897, TN
Eva Hogue b. ca. 1899, TN
One more child born after 1900.
Albert Ross Hogue, a high school teacher, wrote the highly useful reference book, *History of Fentress County*, in 1916. After Matilda's death he remarried, to Josie ---.

Nina Hinds b. 12 Jan 1877, TN
(of Perry & Elizabeth Hinds)

Figure 9: Text snippet from *Delaware Mullinixes* [11].

<Person> → Name [BirthDate, BirthPlace, ChristeningDate, ChristeningPlace, DeathDate, DeathPlace, BurialDate, BurialPlace]

<Marriage> → Name [([$\underline{\text{Spouse}}$, MarriageDate, MarriagePlace])$^+$] STOP?

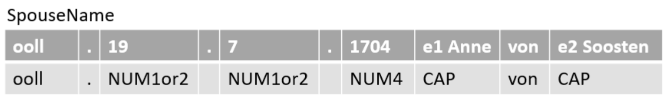
<MultipleSpouseMarriage> → Name [([$\underline{\text{Spouse}}$, MarriageDate, MarriagePlace])$^+$] STOP?

<ParentOf> → Parent (CHILD_GEN$_i$)$^+$

<ChildOf> → Child [Parent1, Parent2]

The grammar notation CHILD_GEN$_i$ in the *ParentOf* grammar denotes any one of the terminals CHILD_GEN$_1$, CHILD_GEN$_2$, ..., CHILD_GEN$_n$ where n is a positive integer. Unlike most grammars, it is possible to write grammatically correct sentences that are semantically meaningless. The parser, however, knows that for sentences to be semantically meaningful, (1) $i = 1$ in the first-encountered CHILD_GEN terminal and (2) after encountering CHILD_GEN$_n$, we must have $i = n$ or $i = n + 1$ or $i < n$. It therefore rejects semantically meaningless records and otherwise builds parse trees as usual. STOP? is an optional terminal symbol used to stop processing "runaway" lists (e.g. child lists) in cases where there is otherwise insufficient context.

These grammars are all regular, type 3 Chomsky grammars. For any inter-generational book like Mullinix, the subscript i on CHILD_GEN$_i$ can be fixed (e.g. $i = 2$ in Fig. 9 and $i = 4$ over pages 80–173, the pages with these inter-generational families in Mullinix [11]). Thus the grammar can be rewritten as:



SpouseName

ooll	.	19	.	7	.	1704	e1 Anne	von	e2 Soosten
ooll	.	NUM1or2		NUM1or2		NUM4	CAP	von	CAP

Figure 10: An Extraction Template using an Example from Fig. 7.

$$\langle\text{ParentOf}\rangle \rightarrow \text{Parent } [\text{CHILD\_GEN}_1, \text{CHILD\_GEN}_2, \text{CHILD\_GEN}_3, \text{CHILD\_GEN}_4]$$

3.4. Real-World Violations of Semi-structured Assumptions

Flögeln's reference identifiers violate the partitioning property of our semi-structured assumption, because text that is not in linear book-text sequence is grafted into extracted records. But the labeled reference identifiers themselves do not violate the partitioning property. It is only in a post-processing step that we replace reference identifiers with the text they reference.

In general, when the linear flow of the text violates our semi-structured requirements, we may nevertheless still be able to process the text. We accommodate violations in two ways: (1) in a post-processing step, substitute referenced out-of-line text for extracted text (e.g. Flögeln reference identifiers), and (2) report parse errors for user correction (e.g. two birth dates for a person, indicating a missed labeling for a record head).

4. Evaluation

We program our extraction engine *by-example* [12, 13]. As Fig. 10 shows, a programmer chooses a text segment containing a token sequence to be extracted (*e1* through *e2*) and labeled (*SpouseName*) such that the segment has sufficient left and sometimes also right context to uniquely classify the extracted text. The text in the example is then abstracted (e.g. the second line in Fig. 10) and generalized (e.g. abstract text patterns generated as a cross product of the set of marriage symbols and the set of common name forms). To form label-value pairs, the lexer matches these abstract patterns to the abstracted book text.

The effort to program the extraction engine depends on the number of examples a user must specify to achieve a desired level of precision and recall. To avoid requiring the programmer to hunt for needed examples, we apply the following strategy: (1) specify all the extraction templates needed for complete coverage of a single typical page and (2) identify for the programmer the examples needed to complete the potentially long tail of additional extraction templates. We identify these examples by noting functional-dependency violations (e.g., no one can have two birth dates), vacuous extractions (e.g., a marriage symbol must have an associated spouse).

We applied this strategy beginning with Flögeln [10], page 15, which has seven households (the first two of which are in Fig. 7). After giving 31 examples, we achieved 100% precision and recall for page 15 and 97% precision and 89% recall against a complete ground truth containing 18,898 facts taken from all 111 pages of families in the Flögeln book. In our application, we are particularly interested in identifying persons—all of them if possible. For person names alone, the recall was 94% with a precision of 99%. Then, following only the system-generated suggestions, we added 45 more examples resulting in 98% precision and 93% recall overall and

97% recall for person names with 99% precision. Although 76 examples may seem like a lot, it is certainly far fewer than 18,898.

In our final step, we are particularly interested for our application in increasing recall for record heads, people for whom vital or relationship information exists. We identify additional needed examples in two ways: (1) recognized household identifiers for which there is no household head and (2) household members for whom we found more than one birth event date or more than one death event date in their information space (e.g., we know that there must be a missed person name if an extracted record has two birth dates for the record-head person). In addition, our ground truth requires names that contain no stray non-name characters. The important point here is that all these names, missing or defective, can be found automatically.

5. Conclusion and Future Work

Although NRR has been studied at length for many years [14], to the best of our knowledge no one except Nagy [15] has taken the approach we present here. Results for the Ely [2], Kilbarchan [9], and Miller [8] books for both our variation and Nagy's [15] are similar (both achieving F-scores above 95%). However, the record compilation algorithm of [15] does not generalize for books like Flögeln [10] because it does not rely on the basic assumption for compiling records that underlies our grammars: for each record type *R*, every compiled relationship among the field values of *R* includes the ontologically committed object of the field value associated with the grammar's first terminal symbol of the production rule that begins with the start symbol (the record-head object described in Section 2).

There are several directions for future work that we would like to pursue. By processing more *Ortsfamilienbücher* (books about families from a specific location, of which the Flögeln text [10] is an example) and other Latin-based language books, we may discover refinements that could improve our process. There are also non-Latin books such as Chinese Jiapu records and books of handwritten records that will likely need different techniques to be developed in order to apply our process and successfully extract structured records. There are also large numbers of layout-based documents such as tables and forms that may be able to be parsed according to ontologically generated grammars if the tables and forms are rewritten first.

We have demonstrated that it is possible to extract large numbers of records from semi-structured text sources with high accuracy and minimal effort and expertise required by following a strategy of parsing using ontologically generated grammars and marking up a relatively small set of examples for each source. Using a conceptual modeling foundation is key to the success of this approach. Because we rely on a well-defined conceptual model and established relational scheme generation and normalization theory, the approach we have presented of parsing based on ontologically generated grammars generalizes for any application where a real-world conceptual model describes the application domain and where data-rich semi-structured text sources are available.

Acknowledgements

We owe a debt of gratitude to our long-time friend and colleague, Emeritus Professor George Nagy (Rensselaer Polytechnic Institute), for blazing the trail to the work presented here.

References

[1] J. Akoka, I. Comyn-Wattiau, S. Lamassé, C. du Mouza, Contribution of conceptual modeling to enhancing historians' intuition - application to prosopography, in: Proceedings of ER 2020, Springer, Vienna, Austria, 2020, pp. 164–173.

[2] G. Vanderpoel, The Ely Ancestry: Lineage of RICHARD ELY of Plymouth, England, The Calumet Press, New York, New York, 1902.

[3] D. Embley, B. Kurtz, S. Woodfield, Object-oriented Systems Analysis: A Model-driven Approach, Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1992.

[4] W. Mok, J. Fong, D. Embley, Generating the fewest redundancy-free XML scheme trees from acyclic conceptual-model hypergraphs in polynomial time, Information Systems 41 (2014) 20–44.

[5] D. Embley, Object Database Development: Concepts and Principles, Addison-Wesley, Reading, Massachusetts, 1998.

[6] N. Chomsky, Three models for the description of language, IRE Transactions on Information Theory 2 (1956) 113–124.

[7] M. Sipser, Introduction to the Theory of Computation, second ed., Thomson Course Technology, Boston, Massachusetts, USA, 2006.

[8] MillerRecords90, Miller Funeral Home Records, 1917 – 1950, Greenville, Ohio, Darke County Ohio Genealogical Society, Greenville, Ohio, 1990.

[9] F. Grant, Index to The Register of Marriages and Baptisms in the Parish of Kilbarchan, 1649–1772, J. Skinner & Company, LTD, Edinburgh, Scotland, 1912.

[10] E. Friedrichs, A. Pech, Familienbuch des Kirchspiels Flögeln: bestehend aus den Dörfern Flögeln und Fickmühlen ; vom Beginn der Kirchenbücher 1700 bis 1900, Deutsche Ortssippenbücher. Reihe A, E. Friedrichs, Bremerhaven, 2000.

[11] M. Blanck, Delaware Mullinixes and their Descendants' Migrations, 1698–1900, Marilyn Mullinix Blanck, Alhambra, California, 2008.

[12] D. Embley, G. Nagy, Green interaction for extracting family information from OCR'd books, in: Proceedings of the 13th IAPR International Workshop on Document Analysis Systems, DAS 2018, IEEE Computer Society, Vienna, Austria, 2018, pp. 127–132.

[13] G. Nagy, Green information extraction from family books, SN Computer Science 1 (2020) 1–23. doi:10.1007/s42979-019-0024-x.

[14] N. Bach, S. Badaskar, A review of relation extraction, 2006. URL: https://www.cs.cmu.edu/~nbach/papers/A-survey-on-Relation-Extraction.pdf.

[15] G. Nagy, Near-perfect relation extraction from family books, in: Proceedings of the International Conference on Document Analysis and Recognition (ICDAR 2021), volume Lecture Notes in Computer Science 12823, Springer Verlag, 2021, pp. 477–491.