Natural Language Understanding for Soft Information Fusion

Stuart C. Shapiro and Daniel R. Schlegel
Department of Computer Science and Engineering
Center for Multisource Information Fusion
and Center for Cognitive Science
University at Buffalo, The State University of New York
Buffalo, New York
{shapiro|drschleg}@buffalo.edu

***Abstract*—Tractor is a system for understanding English messages within the context of hard and soft information fusion for situation assessment. Tractor processes a message through syntactic processors, and represents the result in a formal knowledge representation language. The result is a hybrid syntactic-semantic knowledge base that is mostly syntactic. Tractor then adds relevant ontological and geographic information. Finally, it applies hand-crafted syntax-semantics mapping rules to convert the syntactic information into semantic information, although the final result is still a hybrid syntactic-semantic knowledge base. This paper presents the various stages of Tractor's natural language understanding process, with particular emphasis on discussions of the representation used and of the syntax-semantics mapping rules.**

I. Introduction

Tractor is a system for message understanding within the context of a multi-investigator, multi-university effort on "Hard and Soft Information Fusion" [1]. Information obtained from physical sensors such as RADAR, SONAR, and LIDAR are considered hard information. Information from humans expressed in natural language is considered soft information. Tractor [2] is a computational system that understands isolated English intelligence messages in the counter-insurgency domain for later fusion with each other and with hard information, all to aid intelligence analysts to perform situation assessment. In this context, "understanding" means creating a knowledge base (KB), expressed in a formal knowledge representation (KR) language, that captures the information in an English message.

Tractor takes as input a single English message. The ultimate goal is for Tractor to output a KB representing the semantic information in that message. Later systems of the larger project combine these KBs with each other and with hard information. Combining KBs from different messages and different hard sources is done via a process of data association [1], [3] that operates by comparing the attributes of and relations among the entities and events described in each KB. It is therefore important for Tractor to express these attributes and relations as completely and accurately as possible.

Many systems that are used for the same purpose as Tractor use information extraction techniques. For example, on its web site, Orbis Technologies, Inc. says, "Orbis Technologies, Inc. is a leader in providing cloud computing-based semantic text analytics, using MapReduce, to support *entity extraction,* relationship identification, and semantic search", [1] and information extraction is defined as "the process of identifying within text instances of *specified* classes of entities and of predications involving these entities" [4, emphasis added]. Rather than merely trying to identify certain pre-specified classes of entities and events (people, places, organizations, *etc.*) in a top-down fashion, by looking for them in the text, we want to faithfully identify and describe all the entities and events mentioned in each message in a bottom-up fashion, converting to a semantic representation whatever occurs there.

Our approach is to use largely off-the-shelf software for syntactic processing, to be discussed briefly in §III. The output of syntactic processing is actually a hybrid syntactic-semantic representation, due to the semantic classification information added by named-entity recognizers. We translate the output of the syntactic processing to the KR language we use. The KR language is introduced in §II, and the translator in §IV. This KB is enhanced with relevant ontological and geographical information, briefly discussed in §V. Finally, hand-crafted syntax-semantics mapping rules are used to convert the mostly syntactic KB into a mostly semantic KB. This is still a hybrid syntactic-semantic representation, because the mapping rules do not yet convert all the syntactic information. The specific representation constructs we use are introduced in §VI–VIII. The syntax-semantics mapping rules are discussed in §IX, and some summary information drawn from a semantic KB is shown in §X. Although even the remaining syntactic information in the final KB is useful for data association, our intention is to add mapping rules so that, over time, the KBs that are produced are less syntactic and more semantic. The results of testing and evaluating the system are presented and discussed in §XI.

This paper constitutes an update and current status report on

[1] http://orbistechnologies.com/solutions/cloud-based-text-analytics/ emphasis added.

Tractor, which has been introduced and discussed in a previous set of papers [1], [2], [5], [6], [7]. An overview of the entire Hard and Soft Information Fusion project, and the architecture of the process is given in [1]. An introduction to Tractor and its initial architecture is given in [2]. An introduction to the Context-Based Information Retrieval (CBIR) subprocess of Tractor, its proposed use of spreading activation, and how spreading activation algorithms might be evaluated is given in [6]. A general overview of the role of contextual information in information fusion architectures is given in [5]. Tractor's use of propositional graphs for representing syntactic and semantic information is introduced in [7]. That paper ends with the comment, "The graphs used in this paper have been hand-built using the mappings detailed in section IV. Automating this process to produce propositional graphs such as these is the major implementation focus of future work" [7, p. 527]. That work has now largely been done. This paper reports on the results of that work.

II. SNePS 3

We use SNePS 3 [8] as the KR system for the KBs created by Tractor from the English messages. SNePS 3 is simultaneously a logic-based, frame-based, and graph-based KR system [9], and is the latest member of the SNePS family of KR systems [10]. In this paper, we will show SNePS 3 expressions using the logical notation, $(R\ a_1\ \ldots\ a_n)$, where R is an n-ary relation and $a_1, \ldots, a_n$ are its n arguments. We will refer to such an expression as a "proposition". We will use "assertion" to refer to a proposition that is taken to be true in the KB, and say "assert a proposition" to mean adding the proposition to the KB as an assertion. We will also speak of "unasserting a proposition" to mean removing the assertion from the KB. The arguments of a proposition are terms that could denote words, occurrences of words in the message (called "tokens"), syntactic categories, entities in the domain, events in the domain, classes (also referred to as "categories") of these entities and events, or attributes of these entities and events.

We can classify relations, and the propositions in which they occur, as either: **syntactic**, taking as arguments terms denoting words, tokens, and syntactic categories; or as **semantic**, taking as arguments entities and events in the domain and their categories and properties. A KB is syntactic to the extent that its assertions are syntactic, and is semantic to the extent that its assertions are semantic. The KB first created by Tractor from a message is mostly syntactic. After the syntax-semantics mapping rules have fired, the KB is mostly semantic. A subtle change that occurs as the mapping rules fire is that terms that originally denote syntactic entities are converted into denoting semantic entities.[2]

[2]What we call in this paper the "syntactic KB" and the "semantic KB" were called in other papers the "syntactic propositional graph" and the "semantic propositional graph," respectively. The reason is that, in this paper, we are exclusively using the logic-based view of SNePS 3, whereas in those papers, we used the graph-based view of SNePS 3. Their equivalence is explained in [9].

III. Syntactic Processing

For initial syntactic processing, we use GATE, the General Architecture for Text Engineering [11], which is a framework for plugging in a sequence of "processing resources" (PRs). The most significant PRs we use, mostly from the ANNIE (a Nearly-New Information Extraction System) suite [12], are: the ANNIE Gazetteer, for lexicon-based named-entity recognition; the ANNIE NE Transducer, for rule-based named-entity recognition; the ANNIE Orthomatcher, ANNIE Nominal Coreferencer, and ANNIE Pronominal Coreferencer, for coreference resolution; the GATE Morphological Analyser for finding the root forms of inflected nouns and verbs; the Stanford Dependency Parser, for part-of-speech tagging and parsing; and the GATE Co-reference Editor, for manual corrections of and additions to the results of the three automatic coreference resolution PRs. We added to the lexicons, added some rules to the rule-based PRs, and fixed some program bugs. We did not modify the parser nor the morphological analyser. We can have a person use the Co-reference Editor as part of processing messages, or can process messages completely automatically without using the Co-reference Editor.

The results of GATE processing, with or without the Co-reference Editor, is a set of "annotations", each consisting of an ID, a start and end position within the message's text string, a Type, and a set of attribute-value pairs. Each PR contributes its own set of annotations, with its own IDs, and its own set of attributes and possible values. Only the start and end positions indicate when an annotation of one PR annotates the same text string as an annotation of another PR.

IV. The Propositionalizer

The Propositionalizer examines the annotations produced by the GATE PRs, and produces a set of SNePS 3 assertions. The stages of the Propositionalizer are: annotation merging; correction of minor errors in syntactic categories; canonicalization of dates and times; and processing the structured portion of semi-structured messages. Annotations covering the same range of characters are combined into one SNePS 3 token-denoting term. Dates and times are converted into ISO8601 format. Annotation types, subtypes (where they exist), parts-of-speech, and dependency relations are converted into logical assertions about the tokens. The actual text string of an annotation and the root found by the morphological analyzer are converted into terms and related to the annotation-token by the `TextOf` and `RootOf` relations, respectively. Coreference chains are converted into instances of the SNePS 3 proposition `(Equiv` $t_1 \ldots t_n$`)`, where $t_1 \ldots t_n$ are the terms for the coreferring tokens.

Most of the messages we are dealing with have structured headers, generally consisting of a message number and date, and sometimes a time. A message reporting a call intercept generally lists a description or name of the caller and of the recipient, duration, medium (*e.g., "cell phone" or "text message"*), and intercepting analyst. These are converted into SNePS 3 assertions.

As an example, consider message syn194:

194. 03/03/10 - Dhanun Ahmad has been placed into custody by the Iraqi police and transferred to a holding cell in Karkh; news of his detainment is circulated in his neighborhood of Rashid.

The basic information about the word "placed" in SNePS 3 is

```
(TextOf placed n20)
(RootOf place n20)
(token-start-pos n20 38)
(token-end-pos n20 44)
(SyntacticCategoryOf VBN n20)
```

Here, `n20` is a SNePS 3 term denoting the occurrence of the word "placed" in character positions 38–44 of the message text. The last proposition says that the syntactic category (part-of-speech) of that token is VBN, the past participle of a verb [12, Appendix G].

Some of the dependency information about "placed", with the text to make it understandable is

```
(nsubjpass n20 n169)
(TextOf Ahmad n169)
(prep n20 n22)
(TextOf into n22)
```

That is, "Ahmad" is the passive subject of "placed", and "placed" is modified by a prepositional phrase using the preposition "into".[3]

Some of the information about "Karkh" is[4]

```
(TextOf Karkh n182)
(SyntacticCategoryOf NNP n182)
(Isa n182 Location)
```

Notice that in the first two of these assertions, `n182` denotes a token (a word occurrence), but in `(Isa n182 Location)`, it denotes an entity, specifically a location, in the domain. This change in the denotation of individual constants is a necessary outcome of the fact that we form a KB representing the syntactic information in a text, and then gradually, via the syntax-semantics mapping rules, turn the same KB into a semantic representation of the text.

The SNePS 3 KB that results from the Propositionalizer is what we call the syntactic KB. Although it contains some semantic information, such as `(Isa n182 Location)`, most of the information in it is syntactic.

V. Enhancement

The syntactic KB is enhanced with relevant information of two kinds: ontological taxonomic information is added above the nouns and verbs occurring in the KB; and geographical information is added to geographic place names occurring in the message. The information to be added is found by a process called "Context-Based Information Retrieval" (CBIR) [13].

[3]In a dependency parse, each token actually represents the phrase or clause headed by that token.

[4]Note that we are using `Isa` as the instance relation based on sentences like "*Fido is a dog*". For the subtype (or "subclass") relation we use `Type`.

CBIR looks up each noun and verb in ResearchCyc[5] to find the corresponding Cyc concept(s). Then it adds to the KB the terms above those concepts in OpenCyc.[6]

CBIR also looks up proper nouns in the NGA GeoNet Names Server database,[7] and adds information found there to the KB. For example, the information added about Karkh is

```
(Isa Karkh SectionOfPopulatedPlace)
(Latitude Karkh 33.3217)
(Longitude Karkh 44.3938)
(MGRS Karkh 38SMB4358187120)
```

The information added by CBIR is important to the data association task in deciding when terms from different messages should be considered to be coreferential.

VI. Major Categories of Entities and Events

The actual message texts determine what categories of entities and events appear in the semantic KBs. For example, in the message, "*Owner of a grocery store on Dhubat Street in Adhamiya said ...*", there is a mention of an entity which is an instance of the category store. So the category of stores is represented in the semantic KB. Nevertheless, there are some categories that play a role in the mapping rules in the sense that there are rules that test whether some term is an instance of one of those categories.

Such major categories of entities include: Person; Organization (a subcategory of Group); company; Location; country; province; city; Date; Time; Phone (the category of phone instruments); PhoneNumber (the category of phone numbers); MGRSToken; JobTitle; Dimension (such as age, height, and cardinality); Group (both groups of instances of some category, such as "mosques," and groups of fillers of some role, such as "residents"); ReligiousGroup (such as "Sunni"); and extensionalGroup (a group explicitly listed in a text, such as, "*Dhanun Ahmad Mahmud, Mu'adh Nuri Khalid Jihad, Sattar 'Ayyash Majid, Abd al-Karim, and Ghazi Husayn.*")

Major categories of events include: Action (such as "break" and "search"); ActionwithAbsentTheme (such as "denounce" and "report"); actionWithPropositionalTheme (such as "say" and "hear"); Perception (such as "learn" and "recognize"); and Event itself.

VII. Relations

Relations used in the syntactic and semantic KBs can be categorized as either syntactic relations or semantic relations.

The syntactic relations we use include the following.

- `(TextOf x y)` means that the token y in the message is an occurrence of the word x.
- `(RootOf x y)` means that x is the root form of the word associated with token y.
- `(SyntacticCategoryOf x y)` means that x is the syntactic category (part-of-speech) of the word associated with token y.

[5]http://research.cyc.com/

[6]http://www.opencyc.org/

[7]http://earth-info.nga.mil/gns/html/

- `(r x y)`, where `r` is one of the dependency relations listed in [14], for example `nsubj, nsubjpass, dobj, prep`, and `nn`, means that token `y` is a dependent of token `x` with dependency relation `r`.

The semantic relations we use include the ones already mentioned (such as `Isa` and `Equiv`), and the following.

- `(Type c1 c2)` means that `c1` is a subcategory of `c2`.
- `(hasName e n)` means that `n` is the proper name of the entity `e`.
- `(GroupOf g c)` means that `g` is a group of instances of the class `c`.
- `(GroupByRoleOf g r)` means that `g` is a group of entities that fill the role, `r`.
- `(MemberOf m g)` means that entity `m` is a member of the group `g`.
- `(hasPart w p)` means that `p` is a part of entity `w`.
- `(hasLocation x y)` means that the location of entity `x` is location `y`.
- `(Before t1 t2)` means that time `t1` occurs before time `t2`.
- `(r x y)`, where `r` is a relation (including `possess, knows, outside, per-country_of_birth, org-country_of_headquarters, agent, experiencer, topic, theme, source`, and `recipient`), means that the entity or event `x` has the relation `r` to the entity or event `y`.
- `(a e v)`, where `a` is an attribute (including `cardinality, color, Date, height, Latitude, Longitude, sex, per-religion, per-date_of_birth`, and `per-age`), means that the value of the attribute `a` of the entity or event `e` is `v`.

Two relations, although syntactic, are retained in the semantic KB for pedigree purposes: `(token-start-pos x i)` means that the token `x` occurred in the text starting at character position `i`, and `(token-end-pos x i)` means that the token `x` occurred in the text ending at character position `i`. These are retained in the semantic KBs so that semantic information may be tracked to the section of text which it interprets. Two other syntactic relations, `TextOf` and `RootOf`, are retained in the semantic KB at the request of the data association group to provide term labels that they use for comparison purposes.

We believe that the syntactic relations we use are all that we will ever need, unless we change dependency parsers, or the dependency parser we use is upgraded and the upgrade includes new dependency relations. However, we make no similar claim for the semantic relations.

Assertions that use syntactic relations are called "syntactic assertions," and those that use semantic relations are called "semantic assertions."

VIII. Representation of Events

To represent events, we use a neo-Davidsonian representation [15], in which the event is reified and semantic roles are binary relations between the event and the semantic role fillers. For suggestions of semantic roles, we have consulted the entries at [16]. For example, in the semantic KB Tractor constructed from message syn064,

> 64. 01/27/10 - BCT forces detained a Sunni munitions trafficker after a search of his car netted IED trigger devices. Ahmad Mahmud was placed in custody after his arrest along the Dour'a Expressway, //MGRSCOORD: 38S MB 47959 80868//, in East Dora.

the information about the detain event includes

```
(Isa n18 detain)
(Date n18 20100127)
(agent n18 n16)
(GroupOf n16 force)
(Modifier n16 BCT)
(theme n18 n26)
(Equiv n230 n26)
(Isa n230 Person)
(hasName n230 "Ahmad Mahmud")
```

That is, `n18` denotes a detain event that occurred on 27 January 2010, the agent of which was a group of BCT forces, and the theme of which was (coreferential with) a person named Ahmad Mahmud.

IX. The Syntax-Semantics Mapper

The purpose of the syntax-semantics mapping rules is to convert information expressed as sets of syntactic assertions into information expressed as sets of semantic assertions. The rules were hand-crafted by examining syntactic constructions in subsets of our corpus, and then expressing the rules in general enough terms so that each one should apply to other examples as well.

The rules are tried in order, so that earlier rules may make adjustments that allow later rules to be more general, and earlier rules may express exceptions to more general later rules. As of this writing, there are 147 mapping rules, that may be divided into several categories:

- *CBIR, supplementary enhancement rules* add ontological assertions that aren't in Cyc, but that relate to terms in the message;
- *SYN, syntactic transformation rules* examine syntactic assertions, unassert some of them, and make other syntactic assertions;
- *SEM, semantic transformation rules* examine semantic assertions, unassert some of them, and make other semantic assertions;
- *SYNSEM, true syntax-semantic mapping rules* examine syntactic assertions and maybe some semantic assertions as well, unassert some of the syntactic assertions, and make new semantic assertions;
- *CLEAN, cleanup rules* unassert some remaining syntactic assertions that do not further contribute to the understanding of the message;
- *INFER, inference rules* make semantic assertions that are implied by other semantic assertions in the KB.

Due to space constraints, only a few rules will be discussed.[8]

An example of a syntactic transformation rule is

```
(defrule passiveToActive
  (nsubjpass ?verb ?passsubj)
  =>
  (assert `(dobj ,?verb ,?passsubj))
  (unassert
   `(nsubjpass ,?verb ,?passsubj))
  (:subrule
   (prep ?verb ?bytok)
   (TextOf by ?bytok)
   (pobj ?bytok ?subj)
   =>
   (assert `(nsubj ,?verb ,?subj))
   (unassert `(prep ,?verb ,?bytok))
   (unassert `(pobj ,?bytok ,?subj))))
```

This rule would transform the parse of "*BCT is approached by a man*" to the parse of "*a man approached BCT*". The rule fires even if the "by" prepositional phrase is omitted.

There are also some rules for distribution over conjunctions. One such rule would transform the parse of "*They noticed a black SUV and a red car parked near the courthouse*" to the parse of "*They noticed a black SUV parked near the courthouse and a red car parked near the courthouse*" by adding an additional partmod relation, from the token for "*car*" to the head token of "*parked near the courthouse*". Then another rule would transform that into the parse of "*They noticed a black SUV parked near the courthouse and they noticed a red car parked near the courthouse*" by adding a second dobj relation, this one from the token of "*noticed*" to the token of "*car*."

Some examples of true syntax-semantics mapping rules operating on noun phrases (presented in the relative order in which they are tried) are:

```
(defrule synsemReligiousGroup
  (Isa ?g relig_group_adj)
  (TextOf ?name ?g)
  =>
  (assert `(Isa ,?g ReligiousGroup))
  (assert `(hasName ,?g ,?name))
  (assert '(Type ReligiousGroup Group))
  (unassert `(Isa ,?g relig_group_adj)))
```

This rule would transform the token for "*Sunni*", which the GATE named entity recognizers recognized to name a relig_group_adj, into an entity that is an instance of `ReligiousGroup`, whose name is `Sunni`. It also makes sure that the relevant fact that `ReligiousGroup` is a subcategory of `Group` is included in the semantic KB for the current message.

```
(defrule hasReligion
  (Isa ?religiongrp ReligiousGroup)
  (nn ?per ?religiongrp)
  (hasName ?religiongrp ?religion)
  =>
  (assert (MemberOf ?per ?religiongrp))
  (assert (per-religion ?per ?religion))
  (unassert (nn ?per ?religiongrp)))
```

This rule would assert about the token of "*youth*" in the parse of "*a Sunni youth*" that it is a member of the group named `Sunni`, and that its religion is `Sunni`. It also would unassert the nn dependency of the token of "*Sunni*" on the token of "*youth*".

```
(defrule properNounToName
  (SyntacticCategoryOf NNP ?token)
  (TextOf ?text ?token)
  =>
  (assert `(hasName ,?token ,?text))
  (unassert `(SyntacticCategoryOf
               NNP ,?token))
  (unassert `(TextOf ,?text ,?token)))
```

This rule would transform a token of the proper noun "*Khalid Sattar*" into a token denoting the entity whose name is `"Khalid Sattar"`.

```
(defrule nounPhraseToInstance
  (SyntacticCategoryOf NN ?nn)
  (:when (isNPhead ?nn))
  (RootOf ?root ?nn)
  (:unless (numberTermp ?root))
  =>
  (assert `(Isa ,?nn ,?root))
  (unassert
   `(SyntacticCategoryOf NN ,?nn))
  (unassert `(RootOf ,?root ,?nn)))
```

This rule would transform the token of "*youth*" in the parse of "*a Sunni youth*" into an instance of the category `youth`. The function `isNPhead` returns True if its argument is the head of a noun phrase, recognized by either having a det dependency relation to some token, or by being an nsubj, dobj, pobj, iobj, nsubjpass, xsubj, or agent dependent of some token. (In the corpus we work on, determiners are sometimes omitted.) The `(:unless (numberTermp ?root))` clause prevents a token of a number from being turned into an instance of that number.

Another rule makes the token of a verb an instance of the event category expressed by the root form of the verb. For example, a token of the verb "*detained*" would become an instance of the event category `detain`, which is a subcategory of `Action`, which is a subcategory of `Event`.

Some examples of syntax-semantics mapping rules that analyze clauses (presented in the relative order in which they are tried) are:

```
(defrule subjAction
  (nsubj ?action ?subj)
  (Isa ?action Action)
  =>
```

[8]The rules are shown using the actual rule syntax.

```
  (assert `(agent ,?action ,?subj))
  (unassert `(nsubj ,?action ,?subj)))
```

This rule would make the subject of "*detained*" the agent of a `detain` Action-event.

```
(defrule subjPerception
  (nsubj ?perception ?subj)
  (Isa ?perception Perception)
  =>
  (assert
   `(experiencer ,?perception ,?subj))
  (unassert `(nsubj ,?perception ,?subj)))
```

This rule would make the subject of "*overheard*" the experiencer of a `overhear` Perception-event.

Another rule makes the date of an event either the date mentioned in the dependency parse tree below the event token, for example the date of the capture event in "*Dhanun Ahmad Mahmud Ahmad, captured on 01/27/10, was turned over to ...*" is `20100127`, or else the date of the message being analyzed.

A final set of syntax-semantics mapping rules convert remaining syntactic assertions into "generic" semantic assertions. For example, any remaining prepositional phrases, after those that were analyzed as indicating the location of an entity or event, the "*by*" prepositional phrases of passive sentences, *etc.*, are transformed into a assertion using the preposition as a relation holding between the entity or event the PP was attached to and the object of the preposition.

As syntax-semantics mapping rules convert syntactic information into semantic information, semantic transformation rules move some of that information to their proper places. One example is

```
(defrule repairLatitude
  (Latitude ?name ?lat)
  (hasName ?entity ,?name)
  =>
  (assert (Latitude ?entity ?lat))
  (unassert (Latitude ?name ?lat)))
```

This, and similar, rules move the geographic information shown in §V from the the name of a location to the location itself.

Cleanup rules unassert syntactic assertions that were already converted into semantic assertions, for example unasserting `(TextOf x y)` and `(RootOf x y)` when `(Isa y x)` has been asserted. Other cleanup rules unassert remaining syntactic assertions that do not contribute to the semantic KB, such as the `SyntacticCategoryOf` assertions.

The inference rules make certain derivable assertions explicit for the benefit of the data association operation. For example, the agent of an event that occurred at some location on some date was at that location on that date, and the member of a group g_1 that is a subgroup of a group g_2 is a member of g_2.

X. Results

In order for a person to get an idea of what is in the semantic KBs, we have implemented a simple natural language generation function that expresses the information in a KB in short formalized sentences. Each relation is associated with a sentence frame whose slots are filled in from the relation's arguments. A term with a proper name, or which is coreferential with one with a proper name, is expressed by its name. Otherwise, terms that are instances of some category are expressed by a symbol constructed from its category. For example, some of the information in the semantic KB that Tractor constructed from syn064, shown and discussed in §VIII, is

```
detain18
Instance of: detain
detain18's Date is |20100127|.
detain18 has the relation agent
                 to |BCT forces|.
detain18 has the relation theme
                 to |Ahmad Mahmud|.
detain18 has the relation after
                 to search32.

|BCT forces|
Instance of: Organization
detain18 has the relation agent
                 to |BCT forces|.

search32
Instance of: search
search32's Date is |20100127|.
search32 has the relation theme
                 to car108.
detain18 has the relation after
                 to search32.

|Ahmad Mahmud|
Instance of: (setof Person trafficker)
|Ahmad Mahmud|'s sex is male.
|Ahmad Mahmud|'s Religion is Sunni.
|Ahmad Mahmud| has the relation possess
                       to car108.
|Ahmad Mahmud| is located at Expressway.
|Ahmad Mahmud| is located at Expressway's
                 Date is |20100127|.
detain18 has the relation theme
                 to |Ahmad Mahmud|.
arrest65 has the relation theme
                 to |Ahmad Mahmud|.

arrest65
Instance of: arrest
arrest65's Date is |20100127|.
arrest65 is located at Expressway.
arrest65 has the relation theme
```

```
                 to |Ahmad Mahmud|.
place55 has the relation after
                to arrest65.

place55
Instance of: place
place55's Date is |20100127|.
place55 is located at |East Dora|.
place55 has the relation in
                to custody59.
place55 has the relation after
                to arrest65.

|East Dora|
Instance of: (setof Location
              SectionOfPopulatedPlace)
|East Dora|'s Latitude is |33.2482|.
|East Dora|'s Longitude is |44.4091|.
|East Dora|'s MGRS is 38SMB4496078958.
|East Dora|'s MGRSRadius is |0.5|.
place55 is located at |East Dora|.
```

XI. Evaluation

The mapping rules were developed by testing Tractor on several corpora of messages, examining the resulting semantic KBs, finding cases where we were not happy with the results, examining the initial syntactic KBs, and modifying or adding to the rule set so that an acceptable result was obtained. These "training" messages included: the 100 messages from the Soft Target Exploitation and Fusion (STEF) project [17]; the 7 Bomber Buster Scenario messages [1]; the 13 messages of the Bio-Weapons Thread, 84 messages of the Rashid IED Cell Thread, and 115 messages of the Sunni Criminal Thread, of the 595-message SYNCOIN dataset [18], [19]. None of these messages were actual intelligence messages, but are "a creative representation of military reports, observations and assessments" [19]. Tractor is still a work in progress. We have not yet finished testing, modifying, and adding to the mapping rules using these training sets.

We are currently developing a "grading rubric" to measure the correctness and completeness of the semantic KBs produced by Tractor against manually produced "gold standard" semantic KBs. We will then have to produce those gold standard KBs, and compare them with those produced by Tractor. It is not yet clear whether that comparison could be done automatically, or would require human grading. We hope to report on this grading rubric, and on Tractor's grades in a future paper.

Nevertheless, we can now evaluate how general the mapping rules are, and whether they are perhaps overly general. The generality of the rules will be tested through examination of how often the mapping rules fire on a "test" dataset, not previously examined. We'll look at the amount of syntactic and semantic data there are in the processed graphs from our test and training sets. We'll also look at how many mistakes Tractor makes on the test dataset, to test for over-generality. Combined, these three experiments will show that our rules are general, but not overly so, that the amount of semantic data in the resultant semantic KBs is quite high, and that the degree of semantization compares well with that of our training sets.

We begin by addressing the question of, given that the mapping rules were developed using the training messages, how general are they? To what extent do they apply to new, unexamined, "test" messages? To answer this question, we used the 57 messages of the the Sectarian Conflict Thread (SCT) of the SynCOIN dataset. These messages, averaging 46 words per message, contain human intelligence reports, "collected" over a period of about five months, which describe a conflict among Christian, Sunni, and Shi'a groups. The messages describe events in detail, and entities usually only through their connection to some group or location.

We divided the rules into the six categories listed in §IX, and counted the number of rules used in the SCT corpus, along with the number of rule firings, as seen in Table I. Of the 147 rules currently part of the system, 91 fired during the processing of this corpus for a total of 11,929 rule firings. Sixty-nine rules fired five or more times, and 80 were used in more than one message. 62% of all the rules and 57% of the true syntax-semantics mapping rules fired on the test messages. We conclude that, even though the rules were developed by looking at specific examples, they are reasonably general.

TABLE I
The number of mapping rules in each category, the number of those rules that fired on any message in the SCT dataset, the total number of times those rules fired, and the average number of times they fired per message.

Rule Type	Rule Count	Rules Fired	Times Fired	Firings/msg
CBIR	1	1	474	8.32
SYN	23	13	1,596	28.00
SEM	5	5	328	5.75
SYNSEM	99	56	2,904	50.95
INFER	9	8	135	2.37
CLEAN	10	8	6,492	113.89
TOTAL	147	91	11,929	209.28

The purpose of the syntax-semantics mapping rules is to convert syntactic information about the words, phrases, clauses and sentences in a message into semantic information about the entities and events discussed in the message. We are still in the process of developing the rule set, so it is useful to measure the percentage of each KB that consists of semantic assertions. Table II shows the number of syntactic assertions,[9] the number

TABLE II
For the total SCT dataset, the number of syntactic assertions, the number of semantic assertions and the percent of assertions that are semantic in the syntactic KBs, the semantic KBs, and in the semantic KBs without counting the assertions added by CBIR.

	Syntactic	Semantic	Percent Semantic
Syntactic	2,469	1,149	31.76%
Semantic	538	48,561	98.90%
without CBIR	538	5,646	91.30%

of semantic assertions, and the percent of assertions that are semantic in the initial syntactic KBs, the final semantic KBs, and the final semantic KBs without counting the semantic assertions added by CBIR (*see* §V). The numbers are the totals over all 57 messages of the SCT dataset. As you can see, before the mapping rules, the KBs are almost 70% syntactic, whereas after the mapping rules they are more than 90% semantic. CBIR is purely additive, so it does not reduce the number of syntactic assertions in the KB, but it does increase the semantic content of the KBs to nearly 99%.

The percentage of the semantic KBs from the test message set that is semantic, 91.30%, is very similar to that of the training message sets. For example, the semantic content of the semantic KBs of two of these training sets, the BBS and STEF datasets, are 92.94%, and 94.15%, respectively, as shown in Table III. We conclude that, even though we are still developing the mapping rules, the ones we have so far are converting a large part of the syntactic information into semantic information, and doing so in a way that generalizes from the training sets to test sets.

TABLE III
PERCENT OF THE SEMANTIC KBS WHICH ARE SEMANTIC FOR THE BBS AND STEF TRAINING SETS, EXCLUDING THE CBIR ENHANCEMENTS.

Dataset	Syntactic	Semantic	Pct Semantic
BBS	57	750	92.94%
STEF	517	8,326	94.15%

Since the mapping rules were designed using the training datasets, it is possible that some of the rules that fire in our test dataset (as shown in Table I) are erroneous. That is, the rules may be *too* general. In order to verify that the rules function as expected, we manually verified that the rules were applied only where they should be.

In order to perfrom this experiment we ran the mapping rules on each message in the dataset, noting after each rule firing whether the firing was correct or incorrect. Rules which fired due to misparses earlier in the process were not counted as rules used. A rule was counted as firing correctly if its output was semantically valid and in accord with the intent of the rule.

As Table IV shows, very rarely were rules applied overzealously. Therefore we can say with some certainty that the rules are not only general enough to fire when processing messages from corpora other than the training set, but they are not overly general; the firings produce a valid semantization of the messages.

TABLE IV
THE NUMBER OF RULES USED IN EACH CATEGORY, ALONG WITH THE NUMBER OF TIMES RULES FROM EACH CATEGORY WERE USED IN THE SCT DATASET, AND THE NUMBER OF TIMES THEY WERE USED CORRECTLY.

Rule Type	Rules Used	Times Fired	Fired Correctly	
			Number	Percent
CBIR	1	474	474	100%
SYN	13	1,567	1,548	98.79%
SEM	5	328	328	100%
SYNSEM	56	2,651	2,431	91.7%
INFER	8	85	72	84.7%
CLEAN	8	6,492	6,492	100%
TOTAL	91	11,597	11,345	97.8%

Comparison with Other Systems

Our system produces results which are much different from those of the most related system we're aware of—Orbis Technologies' proprietary Cloud Based Text-Analytics (CTA) software. The output of the two systems are not directly comparable. CTA attempts to identify and find relationships among entities, in the process identifying the entities' types as either Person, Organization, Location, Equipment, or Date. Where we identify all the types of entities (and have more types, such as Group and Event), Orbis only seems to identify them when they are in a relation. An Orbis relation is simple—an entity is `associated with` another entity. Tractor uses a large set of relations for representing complex relationships between entities.

Within the 57 SCT messages, Tractor identified (among many other things) 34 entities which were members of specific groups, the religion of 17 entities, 203 locations of events or entities, and 33 persons or groups with specific roles. It additionally identified 102 agents of specific events, 128 themes of events, and over 125 spatial relationships such as "in", "on" and "near".

XII. CONCLUSIONS

Tractor is a system for message understanding within the context of hard and soft information fusion for situation assessment. Tractor's processing is bottom-up—find whatever is in the text, rather than top-down—look for pre-specified entities, events, and relations. Tractor uses GATE Processing Resources (PRs) for syntactic analysis, including named-entity recognition, coreference resolution, part-of-speech tagging, and dependency parsing. The propositionalizer converts the annotations produced by the GATE PRs into a hybrid syntactic-semantic knowledge base (KB) represented in the SNePS 3 knowledge representation system. Relevant ontological and geographic information is added to the KB, and then hand-crafted syntax-semantics mapping rules convert the syntactic information into semantic information. Although these rules were devised by looking at specific "training" message sets, 62% of them fired on a separate set of "test" messages. Moreover, not counting syntactic information that is used by later stages of fusion, Tractor, operating on the test messages, was found to convert syntactic KBs that are 68% syntactic into semantic KBs that are 91% semantic (99% semantic when added ontological and geographical information is counted). Not counting rule firings on syntactic assertions that resulted from misparsings, 98% of the rule firings on the test messages resulted in semantically correct assertions that were in accord with what the rule was designed to do.

[9]The token position, `TextOf`, and `RootOf` assertions, which are syntactic, but are retained in the semantic KB for pedigree information and to assist in the downstream scoring of entities against ench other, as explained at the end of §VII, have been omitted from the count.

ACKNOWLEDGMENTS

This work has been supported by a Multidisciplinary University Research Initiative (MURI) grant (Number W911NF-09-1-0392) for "Unified Research on Network-based Hard/Soft Information Fusion", issued by the US Army Research Office (ARO) under the program management of Dr. John Lavery.

REFERENCES

[1] G. A. Gross, R. Nagi, K. Sambhoos, D. R. Schlegel, S. C. Shapiro, and G. Tauer, "Towards hard+soft data fusion: Processing architecture and implementation for the joint fusion and analysis of hard and soft intelligence data," in *Proceedings of the 15th International Conference on Information Fusion (Fusion 2012)*. ISIF, 2012, pp. 955–962.

[2] M. Prentice, M. Kandefer, and S. C. Shapiro, "Tractor: A framework for soft information fusion," in *Proceedings of the 13th International Conference on Information Fusion (Fusion2010)*, 2010, p. Th3.2.2.

[3] A. B. Poore, S. Lu, and B. J. Suchomel, "Data association using multiple frame assignments," in *Handbook of Multisensor Data Fusion*, 2nd ed., M. Liggins, D. Hall, and J. Llinas, Eds. CRC Press, 2009, ch. 13, pp. 299–318.

[4] R. Grishman, "Information extraction: Capabilities and challenges," August 2011, notes prepared for the 2011 International Summer School in Language and Speech Technologies, Tarragona, Spain.

[5] J. Gómez-Romero, J. Garcia, M. Kandefer, J. Llinas, J. M. Molina, M. A. Patricio, M. Prentice, and S. C. Shapiro, "Strategies and techniques for use and exploitation of contextual information in high-level fusion architectures," in *Proceedings of the 13th International Conference on Information Fusion (Fusion 2010)*. ISIF, 2010.

[6] M. Kandefer and S. C. Shapiro, "Evaluating spreading activation for soft information fusion," in *Proceedings of the 14th International Conference on Information Fusion (Fusion 2011)*. ISIF, 2011, pp. 498–505.

[7] M. Prentice and S. C. Shapiro, "Using propositional graphs for soft information fusion," in *Proceedings of the 14th International Conference on Information Fusion (Fusion 2011)*. ISIF, 2011, pp. 522–528.

[8] S. C. Shapiro, "An introduction to SNePS 3," in *Conceptual Structures: Logical, Linguistic, and Computational Issues*, ser. Lecture Notes in Artificial Intelligence, B. Ganter and G. W. Mineau, Eds. Berlin: Springer-Verlag, 2000, vol. 1867, pp. 510–524.

[9] D. R. Schlegel and S. C. Shapiro, "Visually interacting with a knowledge base using frames, logic, and propositional graphs," in *Graph Structures for Knowledge Representation and Reasoning*, ser. Lecture Notes in Artificial Intelligence, M. Croitoru, S. Rudolph, N. Wilson, J. Howse, and O. Corby, Eds. Berlin: Springer-Verlag, 2012, vol. 7205, pp. 188–207.

[10] S. C. Shapiro and W. J. Rapaport, "The SNePS family," *Computers & Mathematics with Applications*, vol. 23, no. 2–5, pp. 243–275, January–March 1992, reprinted in [20, pp. 243–275].

[11] H. Cunningham, D. Maynard, K. Bontcheva, V. Tablan, N. Aswani, I. Roberts, G. Gorrell, A. Funk, A. Roberts, D. Damljanovic, T. Heitz, M. A. Greenwood, H. Saggion, J. Petrak, Y. Li, and W. Peters, *Text Processing with GATE (Version 6)*. The University of Sheffield, Department of Computer Science, 2011. [Online]. Available: http://tinyurl.com/gatebook

[12] H. Cunningham, D. Maynard, K. Bontcheva, and V. Tablan, "GATE: A Framework and Graphical Development Environment for Robust NLP Tools and Applications," in *Proceedings of the 40th Anniversary Meeting of the Association for Computational Linguistics (ACL'02)*, 2002.

[13] M. Kandefer and S. C. Shapiro, "An F-measure for context-based information retrieval," in *Commonsense 2009: Proceedings of the Ninth International Symposium on Logical Formalizations of Commonsense Reasoning*, G. Lakemeyer, L. Morgenstern, and M.-A. Williams, Eds. Toronto, CA: The Fields Institute, 2009, pp. 79–84.

[14] M.-C. de Marneffe and C. D. Manning, *Stanford Typed Dependencies Manual*, Stanford University, September 2008, revised for Stanford Parser v. 1.6.9 in September 2011. http://nlp.stanford.edu/software/dependencies_manual.pdf.

[15] T. Parsons, *Events in the Semantics of English: A Study in Subatomic Semantics*. Cambridge, MA: MIT Press, 1990.

[16] *Unified Verb Index*, University of Colorado, 2012, http://verbs.colorado.edu/verb-index/.

[17] K. Sambhoos, J. Llinas, and E. Little, "Graphical methods for real-time fusion and estimation with soft message data," in *Proceedings of the 11th International Conference on Information Fusion (Fusion 2008)*. ISIF, 2008, pp. 1–8.

[18] J. L. Graham, "A new synthetic dataset for evaluating soft and hard fusion algorithms," in *Proceedings of the SPIE Defense, Security, and Sensing Symposium: Defense Transformation and Net-Centric Systems 2011*, 2011, pp. 25–29.

[19] J. L. Graham, J. Rimland, and D. L. Hall, "A COIN-inspired synthetic data set for qualitative evaluation of hard and soft fusion systems," in *Proceedings of the 14th International Conference on Information Fusion (Fusion 2011)*. ISIF, 2011, pp. 1000–1007.

[20] F. Lehmann, Ed., *Semantic Networks in Artificial Intelligence*. Oxford: Pergamon Press, 1992.