Tractor
Toward Deep Understanding of Short Intelligence Messages

Stuart C. Shapiro

Department of Computer Science and Engineering
Center for Cognitive Science,
Center for Multisource Information Fusion
University at Buffalo, The State University of New York
Buffalo, New York 14260
shapiro@buffalo.edu
Acknowledgements

Daniel R. Schlegel, James M. Dobler
Michael Prentice, Paul Bunter, Michael W. Kandefer

Rakesh Nagi, Kedar Sambhoos, Geoff A. Gross, Greg Tauer

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.
Tractor I/O

**Input:** A short English message
- Mostly 1–3 sentences.
- In Counter-insurgency domain.
- Written by human informant or intelligence gatherer.
- Not necessarily “grammatical” English.

**Output:** Semantic Propositional Graph
- Representing information in the message.
- Nodes for Entities, Events, Actions, Categories, Properties, Property Values, Propositions, ...
- Edges represent non-conceptual relations.
**Input:** A short English message
- Mostly 1–3 sentences.
- In Counter-insurgency domain.
- Written by human informant or intelligence gatherer.
- Not necessarily “grammatical” English.

**Output:** Semantic Propositional Graph
- Representing information in the message.
- Nodes for Entities, Events, Actions, Categories, Properties, Property Values, Propositions, ...
- Edges represent non-conceptual relations.
Important Assumption

- A single message is written by a single person at a single time.
- Different messages might be written by different people at different times, without the author of one message being aware of the contents of previous messages.
- Therefore,
  - NLP techniques are appropriate for intra-message coreference resolution,
  - inter-message coreference resolution must be based on semantic descriptions of the various entities.
- Tractor must supply those semantic descriptions.
Important Assumption

- A single message is written by a single person at a single time.
- Different messages might be written by different people at different times, without the author of one message being aware of the contents of previous messages.
- Therefore,
  - NLP techniques are appropriate for intra-message coreference resolution,
  - inter-message coreference resolution must be based on semantic descriptions of the various entities.
- Tractor must supply those semantic descriptions.
Important Assumption

- A single message is written by a single person at a single time.
- Different messages might be written by different people at different times, without the author of one message being aware of the contents of previous messages.
- Therefore,
  - NLP techniques are appropriate for intra-message coreference resolution,
  - inter-message coreference resolution must be based on semantic descriptions of the various entities.
- Tractor must supply those semantic descriptions.
Important Assumption

- A single message is written by a single person at a single time.
- Different messages might be written by different people at different times, without the author of one message being aware of the contents of previous messages.
- Therefore,
  - NLP techniques are appropriate for intra-message coreference resolution,
  - inter-message coreference resolution must be based on semantic descriptions of the various entities.
- Tractor must supply those semantic descriptions.
A single message is written by a single person at a single time.

Different messages might be written by different people at different times, without the author of one message being aware of the contents of previous messages.

Therefore,

- NLP techniques are appropriate for intra-message coreference resolution,
- inter-message coreference resolution must be based on semantic descriptions of the various entities.

Tractor must supply those semantic descriptions.
Introduction

Why Shortness Matters

Since the messages are short, no need to be concerned with

- rhetorical relations
- topic shifts
- etc.
Hard+Soft Information Fusion Architecture
Motivational Example: STEF Messages

- 01/05/07 - Increased hostile sentiment being expressed against U.S. troops by many worshippers outside the al-Anbia mosque in Adhamiya.
- 01/06/07 - Source said a Sunni youth he knows, Khalid Sattar, has become increasingly vocal in denouncing the U.S. at several mosques in Adhamiya.
Motivational Example: STEF Graphs


Questions:

- Relation between “US” and “United States”?
- What was expressed?
- Who denounced whom?
- Does Source know Khalid Sattar?
- Is Khalid Sattar a Sunni youth?
- What did Source say?
- When did these events occur?
Tractor Processing Stream

- **Short English Message**
  - Syntactic Processing
  - Human Coreference Editing
  - XML Syntactic File of Annotations
- Convert XML to SNePS 3
- SNePS 3 Syntactic Propositional Graph
  - Add Contextually Relevant Background/Ontological Information
- Syntax-Semantics Mapping
- SNePS 3 Semantic Propositional Graph
- Express as GraphML
- Semantic Propositional Graph in GraphML Format
Tractor Processing Stream

- **Short English Message**
- **Syntactic Processing**
  - Human Coreference Editing
  - XML Syntactic File of Annotations
  - Convert XML to SNePS 3
  - SNePS 3 Syntactic Propositional Graph
  - Add Contextually Relevant Background/Ontological Information
  - Syntax-Semantics Mapping
  - SNePS 3 Semantic Propositional Graph
  - Express as GraphML
  - Semantic Propositional Graph in GraphML Format
Tractor Processing Stream

- Short English Message
- Syntactic Processing
- Human Coreference Editing
  - XML Syntactic File of Annotations
  - Convert XML to SNePS 3
  - SNePS 3 Syntactic Propositional Graph
  - Add Contextually Relevant Background/Ontological Information
  - Syntax-Semantics Mapping
  - SNePS 3 Semantic Propositional Graph
  - Express as GraphML
  - Semantic Propositional Graph in GraphML Format
Tractor Processing Stream

- Short English Message
- Syntactic Processing
- Human Coreference Editing
- XML Syntactic File of Annotations
  - Convert XML to SNePS 3
  - SNePS 3 Syntactic Propositional Graph
  - Add Contextually Relevant Background/Ontological Information
- Syntax-Semantics Mapping
- SNePS 3 Semantic Propositional Graph
- Express as GraphML
- Semantic Propositional Graph in GraphML Format
Tractor Processing Stream

- Short English Message
- Syntactic Processing
- Human Coreference Editing
- XML Syntactic File of Annotations
- Convert XML to SNePS 3
  - SNePS 3 Syntactic Propositional Graph
  - Add Contextually Relevant Background/Ontological Information
- Syntax-Semantics Mapping
- SNePS 3 Semantic Propositional Graph
- Express as GraphML
- Semantic Propositional Graph in GraphML Format
Tractor Processing Stream

- Short English Message
- Syntactic Processing
- Human Coreference Editing
- XML Syntactic File of Annotations
- Convert XML to SNePS 3
- SNePS 3 Syntactic Propositional Graph
  - Add Contextually Relevant Background/Ontological Information
  - Syntax-Semantics Mapping
- SNePS 3 Semantic Propositional Graph
- Express as GraphML
- Semantic Propositional Graph in GraphML Format
Tractor Processing Stream

- Short English Message
- Syntactic Processing
- Human Coreference Editing
- XML Syntactic File of Annotations
- Convert XML to SNePS 3
- SNePS 3 Syntactic Propositional Graph
- Add Contextually Relevant Background/Ontological Information
  - Syntax-Semantics Mapping
  - SNePS 3 Semantic Propositional Graph
  - Express as GraphML
  - Semantic Propositional Graph in GraphML Format
Tractor Processing Stream

- Short English Message
- Syntactic Processing
- Human Coreference Editing
- XML Syntactic File of Annotations
- Convert XML to SNePS 3
- SNePS 3 Syntactic Propositional Graph
- Add Contextually Relevant Background/Ontological Information
- Syntax-Semantics Mapping
  - SNePS 3 Semantic Propositional Graph
  - Express as GraphML
  - Semantic Propositional Graph in GraphML Format
Tractor Processing Stream

- Short English Message
- Syntactic Processing
- Human Coreference Editing
- XML Syntactic File of Annotations
- Convert XML to SNePS 3
- SNePS 3 Syntactic Propositional Graph
- Add Contextually Relevant Background/Ontological Information
- Syntax-Semantics Mapping
- SNePS 3 Semantic Propositional Graph
- Express as GraphML
- Semantic Propositional Graph in GraphML Format
Tractor Processing Stream

- Short English Message
- Syntactic Processing
- Human Coreference Editing
- XML Syntactic File ofAnnotations
- Convert XML to SNePS 3
- SNePS 3 Syntactic Propositional Graph
- Add Contextually Relevant Background/Ontological Information
- Syntax-Semantics Mapping
- SNePS 3 Semantic Propositional Graph
- Express as GraphML
- Semantic Propositional Graph in GraphML Format
Tractor Processing Stream

- Short English Message
- Syntactic Processing
- Human Coreference Editing
- XML Syntactic File of Annotations
- Convert XML to SNePS 3
- SNePS 3 Syntactic Propositional Graph
- Add Contextually Relevant Background/Ontological Information
- Syntax-Semantics Mapping
- SNePS 3 Semantic Propositional Graph
- Express as GraphML
- Semantic Propositional Graph in GraphML Format
Example Message

1. 01/31/2010, 0700 hrs. – Al Sabah newspaper reports that in response to the new government policy, local presidential candidate Azam Al-Azhar has called for a protest at the Second District Courthouse. Al-Azhar said he would personally attend this protest, and that local residents should expect to see his black SUV arrive at the courthouse at around 1800 hrs.
GATE (General Architecture for Text Engineering)

1. Character Sequence to English Text
   1. ANNIE (a Nearly-New Information Extraction System) Tokenizer
   2. ANNIE Sentence Splitter

2. Morphology
   1. ANNIE POS Tagger
   2. English Stemmer

3. Named Entity Recognition
   1. List-Based "Gazetteer"
   2. ANNIE Rule-Base NE Transducer

4. Coreferencers
   1. ANNIE OrthoMatcher
   2. ANNIE Pronominal Coreferencer

5. Stanford Dependency Parser
GATE (General Architecture for Text Engineering)

1. Character Sequence to English Text
   1. ANNIE (a Nearly-New Information Extraction System) Tokenizer
   2. ANNIE Sentence Splitter

2. Morphology
   1. ANNIE POS Tagger
   2. English Stemmer

3. Named Entity Recognition
   1. List-Based “Gazetteer”
   2. ANNIE Rule-Base NE Transducer

4. Coreferencers
   1. ANNIE OrthoMatcher
   2. ANNIE Pronominal Coreferencer

5. Stanford Dependency Parser
GATE (General Architecture for Text Engineering)

1. Character Sequence to English Text
   - ANNIE (a Nearly-New Information Extraction System) Tokenizer
   - ANNIE Sentence Splitter

2. Morphology
   - ANNIE POS Tagger
   - English Stemmer

3. Named Entity Recognition
   - List-Based “Gazetteer”
   - ANNIE Rule-Base NE Transducer

4. Coreferencers
   - ANNIE OrthoMatcher
   - ANNIE Pronominal Coreferencer

5. Stanford Dependency Parser
GATE (General Architecture for Text Engineering)

1. Character Sequence to English Text
   1. ANNIE (a Nearly-New Information Extraction System) Tokenizer
   2. ANNIE Sentence Splitter

2. Morphology
   1. ANNIE POS Tagger
   2. English Stemmer

3. Named Entity Recognition
   1. List-Based “Gazetteer”
   2. ANNIE Rule-Base NE Transducer

4. Coreferencers
   1. ANNIE OrthoMatcher
   2. ANNIE Pronominal Coreferencer

5. Stanford Dependency Parser
GATE (General Architecture for Text Engineering)

1. Character Sequence to English Text
   1. ANNIE (a Nearly-New Information Extraction System) Tokenizer
   2. ANNIE Sentence Splitter

2. Morphology
   1. ANNIE POS Tagger
   2. English Stemmer

3. Named Entity Recognition
   1. List-Based “Gazetteer”
   2. ANNIE Rule-Base NE Transducer

4. Coreferencers
   1. ANNIE OrthoMatcher
   2. ANNIE Pronominal Coreferencer

5. Stanford Dependency Parser
Syntactic Processing

Results of Tokenizer

1. 01/31/2013, 07:00 hrs. Al-Sabah newspaper reports that in response to the new government policy, a local presidential candidate, Azam Al-Azhar, has called for a protest at the Second District Courthouse. Al-Azhar said he would personally attend this protest and that local residents should expect to see his black SUV arrive at the courthouse at around 1800 hrs.
1. 01/31/2010, 0700 hrs. -- Al Sabah newspaper reports that in response to the new government policy, local presidential candidate Azam Al-Azhar has called for a protest at the Second District Courthouse. Al-Azhar said he would personally attend this protest, and that local residents should expect to see his black SUV arrive at the courthouse at around 1800 hrs.
### Results of POS Tagger & Stemmer

[Image of GATE Developer interface showing POS tagging and annotation sets.]

- **Date:** 28 March 2012
- **Page:** 15/70
- **Author:** S. C. Shapiro (UB)

#### Example Annotation Set

<table>
<thead>
<tr>
<th>Type</th>
<th>Set</th>
<th>Start</th>
<th>End</th>
<th>Id</th>
<th>Features</th>
</tr>
</thead>
<tbody>
<tr>
<td>Token</td>
<td>94</td>
<td>100</td>
<td>1304</td>
<td>4</td>
<td>category=NN, kind=word, length=6, orth=lowercase, stem=policy, string=policy</td>
</tr>
<tr>
<td>Token</td>
<td>100</td>
<td>101</td>
<td>1305</td>
<td>5</td>
<td>category=, kind=punctuation, length=1, stem=, string=,</td>
</tr>
<tr>
<td>Token</td>
<td>102</td>
<td>107</td>
<td>1307</td>
<td>7</td>
<td>category=j, kind=word, length=5, orth=lowercase, stem=local, string=local</td>
</tr>
<tr>
<td>Token</td>
<td>108</td>
<td>120</td>
<td>1309</td>
<td>9</td>
<td>category=h, kind=word, length=12, orth=lowercase, stem=presidential, string=pr</td>
</tr>
<tr>
<td>Token</td>
<td>121</td>
<td>130</td>
<td>1311</td>
<td>11</td>
<td>category=NN, kind=word, length=9, orth=lowercase, stem=candidate, string=can</td>
</tr>
<tr>
<td>Token</td>
<td>131</td>
<td>135</td>
<td>1313</td>
<td>13</td>
<td>category=NNP, kind=word, length=4, orth=upperinitial, stem=azam, string=Azam</td>
</tr>
<tr>
<td>Token</td>
<td>136</td>
<td>139</td>
<td>1315</td>
<td>15</td>
<td>category=NNP, kind=word, length=3, orth=upperinitial, stem=al-, string=Al-</td>
</tr>
<tr>
<td>Token</td>
<td>139</td>
<td>144</td>
<td>1316</td>
<td>16</td>
<td>category=NNP, kind=word, length=5, orth=upperinitial, stem=azhar, string=Azha</td>
</tr>
<tr>
<td>Token</td>
<td>145</td>
<td>148</td>
<td>1318</td>
<td>18</td>
<td>category=VBZ, kind=word, length=3, orth=lowercase, stem=has, string=has</td>
</tr>
<tr>
<td>Token</td>
<td>149</td>
<td>155</td>
<td>1320</td>
<td>20</td>
<td>category=VBN, kind=word, length=6, orth=lowercase, stem=call, string=called</td>
</tr>
<tr>
<td>Token</td>
<td>155</td>
<td>159</td>
<td>1322</td>
<td>22</td>
<td>category=IN, kind=word, length=3, orth=lowercase, stem=for, string=for</td>
</tr>
<tr>
<td>Token</td>
<td>156</td>
<td>161</td>
<td>1324</td>
<td>24</td>
<td>category=DT, kind=word, length=1, orth=lowercase, stem=a, string=a</td>
</tr>
</tbody>
</table>

---

*Note: The interface shows various settings and features related to POS tagging and annotation.*
Results of “Gazetteer-Based” Named Entity Recognition

1. 01/31/2010, 07:00 hrs. – Al Sabah newspaper reports that in response to the new government policy, local presidential candidate Azam Al-Azhar has called for a protest at the Second District Courthouse. Al-Azhar said he would personally attend this protest, and that local residents should expect to see his black SUV arrive at the courthouse at around 1800 hrs.

<table>
<thead>
<tr>
<th>Type</th>
<th>Set</th>
<th>Start</th>
<th>End</th>
<th>Id</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lookup</td>
<td>9</td>
<td>13</td>
<td>1562</td>
<td>{majorType=year}</td>
</tr>
<tr>
<td>Lookup</td>
<td>28</td>
<td>36</td>
<td>1563</td>
<td>{majorType=organization, minorType=newspaper}</td>
</tr>
<tr>
<td>Lookup</td>
<td>31</td>
<td>36</td>
<td>1564</td>
<td>{majorType=person first, minorType=female}</td>
</tr>
<tr>
<td>Lookup</td>
<td>121</td>
<td>130</td>
<td>1565</td>
<td>{majorType=jobTitle}</td>
</tr>
<tr>
<td>Lookup</td>
<td>131</td>
<td>144</td>
<td>1566</td>
<td>{majorType=person full}</td>
</tr>
<tr>
<td>Lookup</td>
<td>136</td>
<td>138</td>
<td>1567</td>
<td>{majorType=person first, minorType=male}</td>
</tr>
<tr>
<td>Lookup</td>
<td>139</td>
<td>144</td>
<td>1568</td>
<td>{majorType=person first, minorType=male}</td>
</tr>
<tr>
<td>Lookup</td>
<td>177</td>
<td>203</td>
<td>1569</td>
<td>{majorType=facility, minorType=building}</td>
</tr>
<tr>
<td>Lookup</td>
<td>184</td>
<td>192</td>
<td>1570</td>
<td>{majorType=loc key, minorType=post}</td>
</tr>
<tr>
<td>Lookup</td>
<td>205</td>
<td>207</td>
<td>1571</td>
<td>{majorType=person first, minorType=male}</td>
</tr>
<tr>
<td>Lookup</td>
<td>208</td>
<td>213</td>
<td>1572</td>
<td>{majorType=person first, minorType=male}</td>
</tr>
<tr>
<td>Lookup</td>
<td>246</td>
<td>250</td>
<td>1573</td>
<td>{majorType=time modifier}</td>
</tr>
<tr>
<td>Lookup</td>
<td>275</td>
<td>284</td>
<td>1574</td>
<td>{majorType=jobTitle pl}</td>
</tr>
<tr>
<td>Lookup</td>
<td>310</td>
<td>315</td>
<td>1575</td>
<td>{majorType=color}</td>
</tr>
<tr>
<td>Lookup</td>
<td>316</td>
<td>319</td>
<td>1576</td>
<td>{majorType=vehicle}</td>
</tr>
</tbody>
</table>
Results of Rule-Based Named Entity Recognition
Results of OrthoMatcher

1. 01/31/2010, 0700 hrs. — Al Sabah newspaper reports that in response to the new government policy, local presidential candidate Azam Al-Azhar has called for a protest at the Second District Courthouse. Al-Azhar said he would personally attend this protest, and that local residents should expect to see his black SUV arrive at the courthouse at around 1800 hrs.
1. 01/31/2010, 0700 hrs. — Al sabah newspaper reports that in response to the new government policy, local presidential candidate Azam Al-Azhari has called for a protest at the Second District Courthouse. Al-Azhari said he would personally attend this protest, and that local residents should expect to see his black SUV arrive at the courthouse at around 1800 hrs.
Result of Dependency Parse

1. 01/31/2010, 07:00 hrs. — Al Sabah newspaper reports that in response to the new government policy, local presidential candidate Azam Al-Azhar has called for a protest at the Second District Courthouse. Al-Azhar said he would personally attend this protest, and that local residents should expect to see his black SUV arrive at the courthouse at around 1800 hrs.
Co-reference Editor
Initial Chain

1. 01/31/2010, 0700 hrs. – Al Sabah newspaper reports that in response to the new government policy, local presidential candidate Azam Al-Azhar has called for a protest at the Second District Courthouse. Al-Azhar said he would personally attend this protest, and that local residents should expect to see his black SUV arrive at the courthouse at around 1800 hrs.
Co-reference Editor
Adding a Chain

1. 01/31/2010, 0700 hrs. — Al Sabah newspaper reports that in response to the new government policy, local presidential candidate Azam Al-Azhari has called for a protest at the Second District Courthouse. Al-Azhari said he would personally attend this protest, and that local residents should expect to see his black SUV arrive at the courthouse at around 1800 hrs.

Add "Second District Courthouse" to [New Chain]

OK Cancel
Co-reference Editor
Adding to a Chain

Add "courthouse" to
Second District Courthouse
Co-reference Editor

Final Chains

Human Coreference Editing

Tractor

28 March 2012 24 / 70
**XML to SNePS 3**

**XML**

```xml
<Annotation Id="510" Type="Token" StartNode="251" EndNode="258">
  <Feature>
    <Name className="java.lang.String">string</Name>
    <Value className="java.lang.String">protest</Value>
  </Feature>
  <Feature>
    <Name className="java.lang.String">category</Name>
    <Value className="java.lang.String">NN</Value>
  </Feature>
  <Feature>
    <Name className="java.lang.String">dependencies</Name>
    <Value className="java.util.ArrayList" itemClassName="java.lang.String">
      det(569)
    </Value>
  </Feature>
  <Feature>
    <Name className="java.lang.String">matches</Name>
    <Value className="java.util.ArrayList" itemClassName="java.lang.Integer">
      482;510
    </Value>
  </Feature>
</Annotation>
```

**SNePS 3**

```prolog
(assert '(token-start-pos 510 251))
(assert '(token-end-pos 510 258))
(assert '(TextOf |protest| 510))
(assert '(SyntacticCategoryOf "NN" 510))
(assert '(det 510 569))
(assert '(Equiv (setof 482 510)))
```
Besides translation, `xml2sneps3`

- combines annotations that cover same substring into one token,
- adjusts message times to GMT,
- adjusts message dates as necessary given time change,
- converts message dates to ISO format.
Adding Contextually Relevant Background/Ontological Information

For each noun in the graph

- Finds the Cyc concept from ResearchCyc
- Loads the ontology above it in OpenCyc into the graph
SNePS 3

SNePS 3 is the latest member of the SNePS Family of KRR systems.

It is still being implemented.

The SNePS 3 KB can be thought of as simultaneously being:

- Logic based,
- Frame based, and
- Graph based.

We have created a user interface which uses all three:

- Assertions and queries of a KB are handled using logic or frames.
- Visualization and inspection is done using propositional graphs.
SNePS 3 GUI

Graph View

Logical Interaction
KB as set of Logical Expressions

The SNePS 3 KB is a set of logical expressions:

- **Atomic terms**
  - Individual constants denoting entities in domain including some relations

- **Arbitrary and indefinite terms** [Shapiro, KR2004]

- **Functional terms**
  - terms denoting atomic propositions
  - terms denoting non-atomic propositions

Use CLIF syntax.

Every logical expression is a term.

Allows propositions about propositions without leaving First-Order.

Internal name of functional terms: $\text{wfti}[!]$

for “well-formed term”.
KB as set of Logical Expressions

The SNePS 3 KB is a set of logical expressions:

- **Atomic terms**
  - Individual constants denoting entities in domain including some relations

- **Arbitrary and indefinite terms** [Shapiro, KR2004]
  - Functional terms including
    - terms denoting atomic propositions
    - terms denoting non-atomic propositions

Use CLIF syntax.
Every logical expression is a term.
Allows propositions about propositions without leaving First-Order.
Internal name of functional terms: \texttt{wfti ![ ! ]}
for “well-formed term”.
KB as set of Logical Expressions

The SNePS 3 KB is a set of logical expressions:

- **Atomic terms**
  - Individual constants denoting entities in domain including some relations

- **Arbitrary and indefinite terms** [Shapiro, KR2004]

- **Functional terms**
  - Including
    - terms denoting atomic propositions
    - terms denoting non-atomic propositions

Use CLIF syntax.

Every logical expression is a term.

Allows propositions about propositions without leaving First-Order.

Internal name of functional terms: \texttt{wfti} [!] for “well-formed term”.
The SNePS 3 KB is a set of logical expressions:

- **Atomic terms**
  - Individual constants denoting entities in domain including some relations

- **Arbitrary and indefinite terms** [Shapiro, KR2004]

- **Functional terms**
  - Including
    - Terms denoting atomic propositions
    - Terms denoting non-atomic propositions

Use CLIF syntax.

Every logical expression is a term.

Allows propositions about propositions without leaving First-Order.

Internal name of functional terms: $wfti$ [!] for “well-formed term”. 
KB as set of Logical Expressions

The SNePS 3 KB is a set of logical expressions:

- **Atomic terms**
  - Individual constants denoting entities in domain including some relations

- **Arbitrary and indefinite terms** [Shapiro, KR2004]

- **Functional terms**
  - Including
    - Terms denoting atomic propositions
    - Terms denoting non-atomic propositions

Use CLIF syntax.
Every logical expression is a term.
Allows propositions about propositions without leaving First-Order.
Internal name of functional terms: \texttt{wfti} [!] for “well-formed term”.
KB as set of Logical Expressions

The SNePS 3 KB is a set of logical expressions:

- **Atomic terms**
  - Individual constants denoting entities in domain including some relations

- **Arbitrary and indefinite terms** [Shapiro, KR2004]

- **Functional terms**
  - including
    - terms denoting atomic propositions
    - terms denoting non-atomic propositions

Use CLIF syntax.

Every logical expression is a term.

Allows propositions about propositions without leaving First-Order.

Internal name of functional terms: \texttt{wft} \texttt{i} [!] for “well-formed term”.
Caseframes

- Based on “The Case for Case” [Fillmore, 1968] and The Berkeley FrameNet Project
  
  [Baker, Fillmore, & Lowe, 1998; Ruppenhofer et al., 2010]

- Frame
  - schematic representation of a situation with a set of participants and conceptual roles.

- Eliminates syntactic differences.

- E.g.
  - Sufian called Ziyad.
  - Ziyad was called by Sufian.
  - a call from Sufian to Ziyad

- We will use “caseframe” for their “frame”

- and use “frame” for an instantiated caseframe.
Caseframes

- Based on “The Case for Case” [Fillmore, 1968] and The Berkeley FrameNet Project [Baker, Fillmore, & Lowe, 1998; Ruppenhofer et al., 2010]

- Frame
  - schematic representation of a situation with a set of participants and conceptual roles.
  - Eliminates syntactic differences.
  - E.g.
    - Sufian called Ziyad.
    - Ziyad was called by Sufian.
    - a call from Sufian to Ziyad

- We will use “caseframe” for their “frame” and use “frame” for an instantiated caseframe.
Caseframes

- Based on “The Case for Case” [Fillmore, 1968] and The Berkeley FrameNet Project
  [Baker, Fillmore, & Lowe, 1998; Ruppenhofer et al., 2010]

- Frame
  - schematic representation of a situation with a set of participants and conceptual roles.

- Eliminates syntactic differences.

- E.g.
  - Sufian called Ziyad.
  - Ziyad was called by Sufian.
  - a call from Sufian to Ziyad

- We will use “caseframe” for their “frame”
- and use “frame” for an instantiated caseframe.
Caseframes

- Based on “The Case for Case” [Fillmore, 1968] and The Berkeley FrameNet Project [Baker, Fillmore, & Lowe, 1998; Ruppenhofer et al., 2010]

- Frame
  - schematic representation of a situation with a set of participants and conceptual roles.

- Eliminates syntactic differences.

- E.g.
  - Sufian called Ziyad.
  - Ziyad was called by Sufian.
  - a call from Sufian to Ziyad

- We will use “caseframe” for their “frame”

- and use “frame” for an instantiated caseframe.
Components of Caseframes

Definition

A caseframe has

- A name
- A sort
- An ordered list of slots
- ...

S. C. Shapiro (UB)
Slots

Slots are defined globally independently of the caseframes that use them.

Definition

A slot has

- A name
- Minimum and maximum number of fillers
- The sort of its fillers
- ...
Examples of Caseframes

Example
Isa is a caseframe of type Proposition with slots member and class.

Example
Call is a caseframe of type Proposition with slots Communicator, Addressee, and Communication.
Frames vs. Logical Terms

- A *frame* is an instance of a caseframe.
- The logical term \( (F, x_1, ..., x_n) \) is represented by an instance of the caseframe named \( F \) whose slots, \( s_1, ..., s_n \) are filled by the representations of \( x_1, ..., x_n \), respectively.
(assert '(Call Sufian Ziyad
    "My brother sends greetings")

creates an instance of the Call caseframe

whose Communicator slot contains the filler Sufian,

whose Addressee slot contains Ziyad,

and whose Communication slot

    contains "My brother sends greetings".

Propositional Graphs

A way of visualizing and traversing the frames.

- Directed Acyclic Graph
- Every term is a node.
  - Individual constants
  - Functional terms (frames)
  - Proposition-denoting functional terms
- Node ID is
  - symbol
  - frame name (\textit{\texttt{wft\_i\_[]}})
- Edges drawn
  from the node corresponding to the frame,
  to the nodes corresponding to the slot fillers
- Edges labeled by slot names
Propositional Graphs

A way of visualizing and traversing the frames.

- **Directed Acyclic Graph**
  - Every term is a node.
    - Individual constants
    - Functional terms (frames)
    - Proposition-denoting functional terms
  - Node ID is
    - symbol
    - frame name ($w_{ft i}[]$)
  - Edges drawn
    from the node corresponding to the frame,
    to the nodes corresponding to the slot fillers
  - Edges labeled by slot names
Propositional Graphs

A way of visualizing and traversing the frames.

- Directed Acyclic Graph
- Every term is a node.
  - Individual constants
  - Functional terms (frames)
  - Proposition-denoting functional terms
- Node ID is
  - symbol
  - frame name \( w_{ft}i[!] \)
- Edges drawn
  from the node corresponding to the frame,
  to the nodes corresponding to the slot fillers
- Edges labeled by slot names
Propositional Graphs

A way of visualizing and traversing the frames.

- Directed Acyclic Graph
- Every term is a node.
  - Individual constants
  - Functional terms (frames)
  - Proposition-denoting functional terms
- Node ID is
  - symbol
  - frame name (wft i [!])
- Edges drawn
  from the node corresponding to the frame,
  to the nodes corresponding to the slot fillers
- Edges labeled by slot names
Propositional Graphs

A way of visualizing and traversing the frames.

- Directed Acyclic Graph
- Every term is a node.
  - Individual constants
  - Functional terms (frames)
  - Proposition-denoting functional terms
- Node ID is
  - symbol
  - frame name ($wft i [!]$)
- Edges drawn
  from the node corresponding to the frame,
  to the nodes corresponding to the slot fillers
- Edges labeled by slot names
Propositional Graphs

A way of visualizing and traversing the frames.

- Directed Acyclic Graph
- Every term is a node.
  - Individual constants
  - Functional terms (frames)
  - Proposition-denoting functional terms
- Node ID is
  - symbol
  - frame name (wft i [])
- Edges drawn
  from the node corresponding to the frame,
  to the nodes corresponding to the slot fillers
- Edges labeled by slot names
Propositional Graphs

A way of visualizing and traversing the frames.

- Directed Acyclic Graph
- Every term is a node.
  - Individual constants
  - Functional terms (frames)
  - Proposition-denoting functional terms
- Node ID is
  - symbol
  - frame name (\texttt{wft} \texttt{[i]})
- Edges drawn
  from the node corresponding to the frame,
  to the nodes corresponding to the slot fillers
- Edges labeled by slot names
Propositional Graphs

A way of visualizing and traversing the frames.

- Directed Acyclic Graph
- Every term is a node.
  - Individual constants
  - Functional terms (frames)
  - Proposition-denoting functional terms
- Node ID is
  - symbol
  - frame name ($wft\ i\ [!]$)
- Edges drawn from the node corresponding to the frame, to the nodes corresponding to the slot fillers
- Edges labeled by slot names
Propositional Graphs

A way of visualizing and traversing the frames.

- Directed Acyclic Graph
- Every term is a node.
  - Individual constants
  - Functional terms (frames)
  - Proposition-denoting functional terms
- Node ID is
  - symbol
  - frame name ($\text{wft}_i [])$
- Edges drawn
  from the node corresponding to the frame,
to the nodes corresponding to the slot fillers
- Edges labeled by slot names
Propositional Graphs

A way of visualizing and traversing the frames.

- Directed Acyclic Graph
- Every term is a node.
  - Individual constants
  - Functional terms (frames)
  - Proposition-denoting functional terms
- Node ID is
  - symbol
  - frame name \( \text{wft} \text{i} [!] \)
- Edges drawn
  from the node corresponding to the frame,
  to the nodes corresponding to the slot fillers
- Edges labeled by slot names
“Sufian, a person in Adhamiya, called Ziyad, a person who, according to Ahmed, is in Ramadi, saying ‘My brother sends greetings.’”
Graph View as Visualization

- Visualized graph is for human comprehension.
- Visualized graph need not be isomorphic to implementation of KB.
- Usefulness of $\texttt{wft}$ nodes:
  - Functional term with more than two arguments (slots).
  - Functional term with more than one filler in a slot.
  - Functional term shown as argument of another (filler in a slot).
- Can show a binary relation with no arc coming into it as a labeled arc (“collapsed arc”).
Graph View as Visualization

- Visualized graph is for human comprehension.
- Visualized graph need not be isomorphic to implementation of KB.
- Usefulness of \( \text{wft} \) nodes:
  - Functional term with more than two arguments (slots).
  - Functional term with more than one filler in a slot.
  - Functional term shown as argument of another (filler in a slot).
- Can show a binary relation with no arc coming into it as a labeled arc ("collapsed arc").
Graph View as Visualization

- Visualized graph is for human comprehension.
- Visualized graph need not be isomorphic to implementation of KB.
- Usefulness of $wft$ nodes:
  - Functional term with more than two arguments (slots).
  - Functional term with more than one filler in a slot.
  - Functional term shown as argument of another (filler in a slot).
- Can show a binary relation with no arc coming into it as a labeled arc (“collapsed arc”).
**Graph View as Visualization**

- Visualized graph is for human comprehension.
- Visualized graph need not be isomorphic to implementation of KB.
- Usefulness of $\text{\texttt{wft}}$ nodes:
  - Functional term with more than two arguments (slots).
  - Functional term with more than one filler in a slot.
  - Functional term shown as argument of another (filler in a slot).
- Can show a binary relation with no arc coming into it as a labeled arc (“collapsed arc”).
Graph View as Visualization

- Visualized graph is for human comprehension.
- Visualized graph need not be isomorphic to implementation of KB.
- Usefulness of wft nodes:
  - Functional term with more than two arguments (slots).
  - Functional term with more than one filler in a slot.
  - Functional term shown as argument of another (filler in a slot).
- Can show a binary relation with no arc coming into it as a labeled arc (“collapsed arc”).
Graph View as Visualization

- Visualized graph is for human comprehension.
- Visualized graph need not be isomorphic to implementation of KB.
- Usefulness of \( wft \) nodes:
  - Functional term with more than two arguments (slots).
  - Functional term with more than one filler in a slot.
  - Functional term shown as argument of another (filler in a slot).
- Can show a binary relation with no arc coming into it as a labeled arc (“collapsed arc”).
Graph View as Visualization

- Visualized graph is for human comprehension.
- Visualized graph need not be isomorphic to implementation of KB.
- Usefulness of $wft$ nodes:
  - Functional term with more than two arguments (slots).
  - Functional term with more than one filler in a slot.
  - Functional term shown as argument of another (filler in a slot).
- Can show a binary relation with no arc coming into it as a labeled arc (“collapsed arc”).
Visualizing a Collapsed Arc

- Slots in a frame are ordered.
  - Order of slots = order of arguments of functional term.
  - Draw collapsed arc from first argument to second argument.

- Name of caseframe = function symbol.
- Label collapsed arc with function symbol.

- Different style of arrow head
  so user knows it’s a collapsed arc.
Visualizing a Collapsed Arc

- Slots in a frame are ordered.
- Order of slots = order of arguments of functional term.
- Draw collapsed arc from first argument to second argument.
- Name of caseframe = function symbol.
- Label collapsed arc with function symbol.
- Different style of arrow head so user knows it’s a collapsed arc.
Visualizing a Collapsed Arc

- Slots in a frame are ordered.
- Order of slots = order of arguments of functional term.
- Draw collapsed arc from first argument to second argument.

- Name of caseframe = function symbol.
- Label collapsed arc with function symbol.

- Different style of arrow head
  so user knows it’s a collapsed arc.
Visualizing a Collapsed Arc

- Slots in a frame are ordered.
- Order of slots = order of arguments of functional term.
- Draw collapsed arc from first argument to second argument.
- Name of caseframe = function symbol.
  - Label collapsed arc with function symbol.
- Different style of arrow head so user knows it’s a collapsed arc.
Visualizing a Collapsed Arc

- Slots in a frame are ordered.
- Order of slots = order of arguments of functional term.
- Draw collapsed arc from first argument to second argument.
- Name of caseframe = function symbol.
- Label collapsed arc with function symbol.

- Different style of arrow head so user knows it’s a collapsed arc.
Visualizing a Collapsed Arc

- Slots in a frame are ordered.
- Order of slots = order of arguments of functional term.
- Draw collapsed arc from first argument to second argument.

- Name of caseframe = function symbol.
- Label collapsed arc with function symbol.

- Different style of arrow head so user knows it’s a collapsed arc.
Example of Collapsed Graph: Before

The uncollapsed version of Suifian calling Ziyad example:
Example of Collapsed Graph: After

The collapsed version of Suifian calling Ziyad example:
SNePS 3 Syntactic Graph Visualized (Uncollapsed)
SNePS 3 Syntactic Graph Visualized (Collapsed)
Mapping Rule: properNounToName
Mapping Rule: nounPhraseToInstance
Mapping Rule: \textit{atLocation}
Mapping Rule: colorProperty
Mapping Rule: distribVerbOverConj
Mapping Rule: distribSubjOverXcomp
7. 01/31/2010, 1817 hrs. – Blue team reports that a medium height man with dark hair just entered a blue car by the Second District Courthouse. The man was wearing a tan jacket. They are not sure where he came from.
Mapping Rules:
nounPhraseToInstance; madeOfSubstance; colorProperty; hasDimensionValue; withTypicalPart
CBIR Productivity

Over the 7 messages of the Bomber Buster data set
Noun types looked up: 80
Noun tokens: 165
Assertions added to graphs: 11,860
## Mapping Rule Usage

<table>
<thead>
<tr>
<th>Rule</th>
<th>bbs1</th>
<th>bbs2</th>
<th>bbs3</th>
<th>bbs4</th>
<th>bbs5</th>
<th>bbs6</th>
<th>bbs7</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>addRelevantOntology</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>8</td>
</tr>
<tr>
<td>substringCoreference</td>
<td>10</td>
<td>4</td>
<td>12</td>
<td>2</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>46</td>
</tr>
<tr>
<td>distribModOverConj</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>properNounToName</td>
<td>13</td>
<td>7</td>
<td>15</td>
<td>3</td>
<td>6</td>
<td>4</td>
<td>4</td>
<td>52</td>
</tr>
<tr>
<td>nounPhraseToInstance</td>
<td>5</td>
<td>9</td>
<td>4</td>
<td>12</td>
<td>4</td>
<td>2</td>
<td>5</td>
<td>41</td>
</tr>
<tr>
<td>madeOfSubstance</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>atLocation</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>3</td>
</tr>
<tr>
<td>modLocation</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>colorProperty</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>2</td>
<td>0</td>
<td>3</td>
<td>13</td>
</tr>
<tr>
<td>hasDimensionValue</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>withTypicalPart</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>distribVerbOverConj</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>3</td>
</tr>
<tr>
<td>passiveToActive</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>distribSubjOverXcomp</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>3</td>
</tr>
<tr>
<td>isaIsEquiv</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>3</td>
</tr>
<tr>
<td>prepToRelation</td>
<td>1</td>
<td>3</td>
<td>0</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>7</td>
</tr>
<tr>
<td>removeRedundantStems</td>
<td>56</td>
<td>59</td>
<td>37</td>
<td>73</td>
<td>28</td>
<td>17</td>
<td>40</td>
<td>310</td>
</tr>
<tr>
<td>removeTextWhenIsa</td>
<td>6</td>
<td>12</td>
<td>4</td>
<td>13</td>
<td>4</td>
<td>2</td>
<td>5</td>
<td>46</td>
</tr>
<tr>
<td>removeTextWhenMadeOf</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>98</strong></td>
<td><strong>101</strong></td>
<td><strong>77</strong></td>
<td><strong>113</strong></td>
<td><strong>54</strong></td>
<td><strong>33</strong></td>
<td><strong>69</strong></td>
<td><strong>545</strong></td>
</tr>
</tbody>
</table>
Timings

Average running time per message
Over the 7 messages of the Bomber Buster data set
On a Core i7 2600K @ 3.4ghz, with 16GB RAM

GATE (without human coreference editing): 2.642 secs
xml2sneps3: 0.978 secs
Propositionalizer (without CBIR): 0.596 secs
Propositionalizer (with CBIR): 74.547 secs

(Significant start-up time, so steady-state is faster.)
Motivational Example: STEF Messages

- 01/05/07 - Increased hostile sentiment being expressed against U.S. troops by many worshippers outside the al-Anbia mosque in Adhamiya.
- 01/06/07 - Source said a Sunni youth he knows, Khalid Sattar, has become increasingly vocal in denouncing the U.S. at several mosques in Adhamiya.
Motivational Example: STEF Graphs

[Image of graph]

Questions:

- Relation between “US” and “United States”?
- What was expressed?
- Who denounced whom?
- Does Source know Khalid Sattar?
- Is Khalid Sattar a Sunni youth?
- What did Source say?
- When did these events occur?

Performance on Motivational Example

Tractor on STEF002

S. C. Shapiro (UB) 28 March 2012 59 / 70
Performance on Motivational Example

STEF001: “US”

01/05/07: Increased hostile sentiment being expressed against troops by many worshippers outside the mosque in Adhamiya, U.S.
Performance on Motivational Example

STEF001: What was Expressed?

Graph View: stef001.sneps

Semantic Types
- Entity
  - Act
  - Policy
  - Proposition
  - Thing
  - Action
  - Category

Caseframes
- and
- and/or
- attribute
- Being_named
- DepGraphOf
- Eqv
- Full-Name
- has_location

Contexts
- Contexts
  - BaseCT
  - DefaultCT

S. C. Shapiro (UB)
Tractor
28 March 2012 62 / 70
STEF002: Who denounced whom?
STEF002: Does Source know Khalid Sattar?
STEF002: Is Khalid Sattar a Sunni youth?
STEF002: What did Source say?

01.06.07 — Source said a Sunni youth he knows, Khalid Sattar, has become increasingly vocal in denouncing the ... at several mosques in Adhamiya, U.S.
STEF001: When did these events occur?
STEF002: When did these events occur?
Summary

- Short English message, not necessarily “grammatical”.
- Syntactic processing within GATE.
- Named Entity Recognition, List-Based and Rule-Based.
- Automatic Coreferencers.
- Human Coreference Editing.
- Dependency Parser.
- Syntactic Propositional Graph: Syntactic Relations.
- Syntax-Semantics Mapping Rules.
- Semantic Propositional Graph reflects information in message.
Summary

- Short English message, not necessarily “grammatical”.
- Syntactic processing within GATE.
- Named Entity Recognition, List-Based and Rule-Based.
- Automatic Coreferencers.
- Human Coreference Editing.
- Dependency Parser.
- Syntactic Propositional Graph: Syntactic Relations.
- Syntax-Semantics Mapping Rules.
- Semantic Propositional Graph reflects information in message.
Summary

- Short English message, not necessarily “grammatical”.
- Syntactic processing within GATE.
- Named Entity Recognition, List-Based and Rule-Based.
  - Automatic Coreferencers.
  - Human Coreference Editing.
  - Dependency Parser.
- Syntactic Propositional Graph: Syntactic Relations.
- Syntax-Semantics Mapping Rules.
- Semantic Propositional Graph reflects information in message.
Summary

- Short English message, not necessarily “grammatical”.
- Syntactic processing within GATE.
- Named Entity Recognition, List-Based and Rule-Based.
- Automatic Coreferencers.
  - Human Coreference Editing.
  - Dependency Parser.
  - Syntactic Propositional Graph: Syntactic Relations.
  - Syntax-Semantics Mapping Rules.
  - Semantic Propositional Graph reflects information in message.
Summary

- Short English message, not necessarily “grammatical”.
- Syntactic processing within GATE.
- Named Entity Recognition, List-Based and Rule-Based.
- Automatic Coreferencers.
- Human Coreference Editing.
- Dependency Parser.
- Syntactic Propositional Graph: Syntactic Relations.
- Syntax-Semantics Mapping Rules.
- Semantic Propositional Graph reflects information in message.
Summary

- Short English message, not necessarily “grammatical”.
- Syntactic processing within GATE.
- Named Entity Recognition, List-Based and Rule-Based.
- Automatic Coreferencers.
- Human Coreference Editing.
- Dependency Parser.
  - Syntactic Propositional Graph: Syntactic Relations.
  - Syntax-Semantics Mapping Rules.
  - Semantic Propositional Graph reflects information in message.
Summary

- Short English message, not necessarily “grammatical”.
- Syntactic processing within GATE.
- Named Entity Recognition, List-Based and Rule-Based.
- Automatic Coreferencers.
- Human Coreference Editing.
- Dependency Parser.
- Syntactic Propositional Graph: Syntactic Relations.
  - Syntax-Semantics Mapping Rules.
  - Semantic Propositional Graph reflects information in message.
Summary

- Short English message, not necessarily “grammatical”.
- Syntactic processing within GATE.
- Named Entity Recognition, List-Based and Rule-Based.
- Automatic Coreferencers.
- Human Coreference Editing.
- Dependency Parser.
- Syntactic Propositional Graph: Syntactic Relations.
- Syntax-Semantics Mapping Rules.
- Semantic Propositional Graph reflects information in message.
Summary

- Short English message, not necessarily “grammatical”.
- Syntactic processing within GATE.
- Named Entity Recognition, List-Based and Rule-Based.
- Automatic Coreferencers.
- Human Coreference Editing.
- Dependency Parser.
- Syntactic Propositional Graph: Syntactic Relations.
- Syntax-Semantics Mapping Rules.
- Semantic Propositional Graph reflects information in message.
Further Reading

