A Uniform Architecture for Parsing and Generation of Natural Language

Günter Neumann
DFKI GmbH
66123 Saarbrücken
neumann@dfki.de
Overview

Work based on Neumann:94 (Ph.D. thesis), Neumann:98 (AIJ)

1. Parsing and Generation

2. Results

3. Motivation

4. State of the Art

5. A New Uniform Architecture

6. Parsing and Generation: UTA can do both

7. Interleaving of Parsing and Generation

8. Conclusion and Future Direction
Uniform grammatical processing

• Parsing: given a string, compute all possible logical forms (wrt. the given grammar)

• Generation: given a logical form, compute all possible strings

• Uniformity
  – use of one and the same grammar for performing both tasks
    $\Rightarrow$ reversible grammar
  – use of the same algorithm
    $\Rightarrow$ uniform algorithm
Results

- **Uniform Tabular Algorithm (UTA):**
  - constraint-based grammars
  - generalized Early deduction
  - flexible agenda mechanism
  - On-line
  - Input as essential feature
    * dynamic selection function
    * uniform chart mechanism
  $\implies$ uniform and task-oriented processing

- **Performanz model on the basis of a uniform architecture:**
  - item-sharing between parsing and generation
  - incremental self-monitoring/revison strategies
  - generation of un-ambiguous strings
  - generation of paraphrases
  - any-time mode
  $\implies$ interleaved parsing and generation

- **Implementation in Common Lisp and CLOS**
Uniform grammatical processing

Why uniform grammatical processing?

• Theoretical:
  – Occam’s razor
  – psycho-linguistic motivations

• Practical:
  – reduced redundancies
  – simpler consistency tests
  – knowledge acquisition
  – compact and modular systems

• Application:
  – grammar development
  – interactive grammar-/style-checker
  – incremental text processing
  – monitoring and revision
  – generation of paraphrases
  – processing of elliptic expressions
  – combination of learning-/ preference-based methods
  – …
Reversible grammars

- Language as a relation $R$:
  wellformed strings $\times$ logical forms ($R \subseteq S \times LF'$)
- Parsing: $s$, compute $\{lf_i | < s, lf_i > \in R\}$
- Generation: $lf$, compute $\{s_i | < s_i, lf > \in R\}$
- Reversible grammar: define $R$ with one grammar
- Ambiguity and paraphrases

Lösche das Verzeichnis mit den Systemtools!
Uniform grammatical processing

Current state of the art

Type A

Source Grammar

Parsing Grammar

Sem. Expr.

Parser

String

Generation Grammar

Sem. Expr.

Generator

String

Type B

Source Grammar

Sem. Expr.

Parser

String

Generator

String

Type C

Source Grammar

Parsing Grammar

Sem. Expr.

Uniform Algorithm

String

Generation Grammar

Type D

Source Grammar

Sem. Expr.

Uniform Algorithm

String
Disadvantages of current models

• Types A, B, C
  – approaches: Block (A), Strzalkowski (C), Dymetman et al. (C)
  – high degree of redundancies (A, C)
  – testing of source grammar not possible (A, C)
  – interleaved parsing and generation not meaningful

• Type D
  – approaches: Shieber, van Noord, Gerdeman
  – interleaved approach possible
  – poor dynamic of the models
  – parsing-oriented chart
  – restricted view on uniformity
A New Uniform Model
Uniform grammatical processing

Constraint-based grammars

- e.g., LFG, HPSG, CUG

- Reversibility
  - uniform representation (PHON, SYN, SEM)
  - word, phrase, clause level
  - structure sharing
  - declarative

Example:

```
| cat   | sentence               |
| phon  | (peter, cries)        |
| syntax| ...                   |
| dtrs  |                       |
| phon  |                       |
| syntax|                       |
| semantics | Arg rel the-peter' |
| semantics | Sem |                     |

| cat | noun peter |
| phon |   |
| syntax | agr [per 3 num sg] |
| semantics |                   |

| cat | verb cries |
| phon |   |
| syntax | agr [per 3 num sg] |
| semantics |                   |
| semantics | rel cry arg Arg |
```

Günter Neumann

DFKI
Constraint Logic programming CLP

- Generalization of conventional logic programming to arbitrary constraint-languages (Höffeld&Smolka:88)

- Representation of grammar as definite clauses
  - rule: \( q \leftarrow p_1, \ldots, p_n, \phi \)
  - lexical element: \( q \leftarrow \phi \)

- Goal-reduction rule:
  \[
  \text{goal: } p_1, \ldots, p(\vec{x}), \ldots p_n, \phi \\
  \text{clause: } p(\vec{x}) \leftarrow q_1, \ldots, q_m, \psi \\
  \implies \\
  \text{new goal: } p_1, \ldots, q_1, \ldots, q_m, \ldots, p_n, \phi, \psi
  \]

- Constraint-solver: \( \text{unify}(\phi, \psi) \)

- Parsing and generation: queries of form \( \leftarrow q, \phi \)
UTA: A Uniform Algorithm for Parsing and Generation

• Goal: uniform and task-oriented Processing

• Uniform control logic: generalized Earley deduction (based on Pereira&Warren:83)
  – grammar (rule, lexicon), item sets
  – item: lemma with selected element (SEL)
  – active item (AI): \( h \leftarrow b_0 \ldots b_n ; i ; idx \)
  – passive item (PI): \( h \leftarrow \epsilon ; \epsilon ; idx \)
  – blocking-test: subsumption
UTA: A Uniform Algorithm for Parsing and Generation

• Inference rules:
  – prediction: \texttt{ABSTR(SEL(AI)) unify head(rule)}
  – completion: \texttt{AI minus SEL}
    * scanning: \texttt{SEL(AI) unify lexical element}
    * active completion: \texttt{SEL(AI) unify PI}
    * passive completion: \texttt{PI unify SEL(AI)}

• New clauses (Items): determine \texttt{SEL} using dynamic selection function \texttt{SF}

Prediction: \texttt{⟨Φ[Rule]; SF(Φ[Rule], EF); Idx⟩}
Uniform grammatical processing

Parametrization of UTA

• Relevant parameter: Essential Feature $EF \rightarrow$
  the feature, that carries the input (e.g., PHON or SEM)
    – parametrized selection function
      * $EF$ guides ordering of processing of rhs(rule)

    – paramertized item set
      * $EF$ used for defining equivalence classes

• Parsing and generation with UTA
  $\Rightarrow$ main difference is the different input structure
Uniform grammatical processing

**Parametrizable selection function**

- Choose that element, whose Essential Feature is instantiated, else take the left-most one

- Implications:
  - data-driven selection, e.g.,
    * left-to-right (e.g., for parsing)
    * functor-first (e.g., for generation)
    * or both
    * integration of preferences
  - grammar itself has influence on control

Beispiel:

\[
\text{sign} \left( \begin{array}{c}
\text{cat: } \text{vp} \\
\text{sc: } \text{Tail} \\
\text{sem: } \text{Sem} \\
\text{lex: } \text{no} \\
\text{v2: } \text{V} \\
\text{phon: } \text{P}_0-\text{P}
\end{array} \right) \leftarrow \text{sign} \left( \text{Arg} \begin{array}{c}
\text{phon: } \text{P}_0-\text{P}_1
\end{array} \right), \text{sign} \left( \begin{array}{c}
\text{cat: } \text{vp} \\
\text{sc: } \langle \text{Arg|Tail} \rangle \\
\text{sem: } \text{Sem} \\
\text{v2: } \text{V} \\
\text{phon: } \text{P}_1-\text{P}
\end{array} \right)
\]
Structured item set

- Idea: divide item set into equivalence classes

- Determination of equivalence classes by means of Essential Feature

  \[ \text{item set is structured according to input structure, e.g.,} \]
  \[ \text{as sequence in case of parsing} \]
  \[ \text{as functor/argument tree in case of generation} \]
  \[ \text{set in case of MRS} \]

- Advantage:
  - application of inference rules on subsets
  - blocking-test only on subsets
  - on-the-fly creation

- Details:
  - item set: \( \langle AI, PI, Idx \rangle \)
  - \( \forall \text{ items: } \text{EF compatible} \rightarrow Idx \)
  - PI: EF of Head, AI: EF of SEL
Flexible agenda mechanism

- Guides order of processing of new items
- Sorts items according to preference
- Activation of clauses and insertion into item set according to preference
- Advantage:
  - depth-first, breadth-first, best-first, random
  - blocking-test only on “activated” clauses
  - interleaved parsing and generation: different preference rules
Uniform grammatical processing

Parsing and Generation

- Parsing and generation as queries with instantiated essential feature

- Parsing:
  \[ \text{sign} \left( \text{phon} \left( \text{heute, erzählt, peter, lügen}\right) \right) \]

- Generation:
  \[ \text{sign} \left( \text{sem} \left( \text{mod} \left( \text{heute} \right) \right) \left( \text{pred} \left( \text{erzählen} \right) \right) \left( \text{arg1} \left( \text{peter} \right) \right) \left( \text{arg2} \left( \text{lügen} \right) \right) \right) \]

- EF-proof problem (s.a. VanNoord:93):
  Value of \( \text{EF}_{\text{Query}} \) = Value of \( \text{EF}_{\text{Answer}} \)
### Parsing of “sieht Peter mit Maria”

#### Uniforme Verarbeitung natürlicher Sprache

<table>
<thead>
<tr>
<th>Agenda</th>
<th>Current Task</th>
<th>Item of alternative</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1(vp ← v np pp; 0)</td>
</tr>
<tr>
<td>1,2</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>1,3</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>1,4,5</td>
<td>5</td>
<td>4(np ← np pp; 0)</td>
</tr>
<tr>
<td>1,4</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>1,6,7</td>
<td>7</td>
<td>6(np ← np pp; 0)</td>
</tr>
<tr>
<td>1,6,8</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>1,6,9</td>
<td>9</td>
<td></td>
</tr>
<tr>
<td>1,6,10</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>1,6,11</td>
<td>11</td>
<td></td>
</tr>
<tr>
<td>1,6,12,13</td>
<td>13</td>
<td>12(np ← np pp; 0)</td>
</tr>
<tr>
<td>1,6,14</td>
<td>14</td>
<td></td>
</tr>
<tr>
<td>1,6,15</td>
<td>15</td>
<td></td>
</tr>
<tr>
<td>1,6,16</td>
<td>16</td>
<td></td>
</tr>
<tr>
<td>1,6,17</td>
<td>17</td>
<td></td>
</tr>
<tr>
<td>1,6,18</td>
<td>18 First Result</td>
<td></td>
</tr>
<tr>
<td>1,6,19</td>
<td>12</td>
<td></td>
</tr>
<tr>
<td>1,6</td>
<td>6</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>17</td>
<td></td>
</tr>
<tr>
<td>1,6</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>1,6,19</td>
<td>19</td>
<td></td>
</tr>
<tr>
<td>1,6,20</td>
<td>20</td>
<td></td>
</tr>
<tr>
<td>1,6,21</td>
<td>21</td>
<td></td>
</tr>
<tr>
<td>22</td>
<td>22 Second Result</td>
<td></td>
</tr>
</tbody>
</table>

Günter Neumann

DFKI
Uniform grammatical processing

Generation of “sehen(Peter,mit(Maria))”

Günter Neumann  
DFKI
Interleaving of parsing and generation

• Consider parsing and generation not in isolation
  
  – use of partial results of the other direction
    \[ \Rightarrow \text{item-sharing between parsing \& generation} \]

  – use of one direction
    as additional control of the other
    \[ \Rightarrow \text{incremental self-monitoring} \]

• Uniform algorithm required
Item-Sharing

- Idea: exchange of partial results between parsing and generation

  \[\Rightarrow\]

  computed passive items of one direction are automatically provided for the other direction

- Parsing and generation use same passive items \(\Rightarrow\) item-sharing

- Advantage: re-use of partial results for parsing and generation
Item sharing: architecture
Monitoring and revision during generation

- Psycholing. motivation (cf. Levelt:89)

- Here: avoid mis-understandings

  “ Lösche das Verzeichnis mit den Systemtools”
  “ Lösche mit den Systemtools das Verzeichnis”

- Generation of paraphrases $\Rightarrow$ interactive desambiguation

  “Do you mean: X or Y ?”

- Problem:
  choice between possible paraphrases
Monitoring and revision: Idea

- ‘Revise’ relevante structures of an ambiguous utterance

  $\Rightarrow$ parsing to recover ambiguities

- Assumption: It is possible to revise an utterance locally in order to generate an un-ambiguous utterance with the same meaning

- Neumann und VanNoord 92, non-incremental algorithm
Uniform grammatical processing

Incremental Monitoring and revision

• If local, then revision applicable on partial structures $\implies$ incremental

  – Removing the folder with the system tools can be very dangerous.
  – Visiting relatives can be boring.
  – Visiting relatives are boring.
  – During the ball I danced with a lot of people.
  – I know of no better ball.
Problems for incremental self-monitoring

• When should ambiguity check take place?

• For which just generated partial string $\alpha$ should revision take place?

$\Rightarrow$ Lookback(n)-strategy:

– parse $\alpha$ relative to adjacent already generated partial string $\beta$

– if $\beta$ does not exist, or $\beta\alpha$ not parsable or not ambiguous, then no revision
Fundamental strategy

- Generate/Parse/Revise feedback-loop:
  - during generation for just generated substring $\alpha$
  
  - monitoring: parse $\alpha$ relative to context

  - revision: if ambiguous, then revision

  
  “[Lösche][das Verzeichnis]...[mit den S.t.]”
  $\Rightarrow$ parse
  “das Verzeichnis mit den S.t.”
  $\Rightarrow$ revise
  “[Lösche][mit den S.t.][das Verzeichnis]”
p-completion($\pi$) is:

For every active item $A_i \in I_{idx}$:

if $\Phi = \text{UNIFY}(\text{sel}(A_i), h)$ and $\Phi \neq \text{fail}$ then

if NOT($\text{AND}(\text{Monitor}_?, \text{REVISION}-\text{P}(\Phi[A_i], \pi))$) then

with reduced lemma $R_l = \Phi[A_i - \text{sel}(A_i)]$ do

... 

od

revision-\text{p}(A_i, \pi) is:

with $\text{ExtendedString} = \text{GET-CONTEXT}(A_i, \pi, n)$;

if $\text{ExtendedString}$ then

with $\text{ParseRes} = \text{PARSE}(\text{ExtendedString})$;

if $\text{AND}(\text{ParseRes}, \text{AMBIGUOUS}(A_i, \text{ParseRes}))$

then true else false fi

else false fi.
Incremental self-monitoring with UTA

• Self-embedded control of generation module

• Agenda mechanism automatically realizes revision

• Revision: prune possible set of answers

• Chart-based incremental interleaved parsing & generation

• Specific control:
  – any-time mode
  – lookback(n)-strategy
  – use of grammar-specific information can easily be integrated (preferences, no complements, subset of adjuncts)
Uniform grammatical processing

Conclusion: the main results

• Novel: NLP on the basis of interleaved parsing and generation

• Theoretical: competence-based performance model

• Practical: uniformity as application impact

• For whole NLP:
  – self-monitoring/self-control \(\Rightarrow\)
    * flexibility
    * robustness
    * adaptibility
Possible future directions

- Incremental generation with MRS
- Integration of Machine Learning
  - Explanation-based Learning (EBL):
    automatic computation of prototypical constructions (templates)
  - UTA: reversible EBL, Template-sharing
  - principle-guided induction of reversible grammars

- Preference-based methods:
  - stochastic lexicalized tree grammars
    (NeumannFlickinger:99)
  - hearer-adaptable monitoring/revision

- bidirectional dialog systems