Applying Transition Networks in Translating Polish E-Mails

Krzysztof Jassem, jassem@amu.edu.pl, Filip Graliński filipg@amu.edu.pl
Tomasz Kowalski chiro@interia.pl
Adam Mickiewicz University, Faculty of Mathematics and Computer Science, Poznań, Poland

Summary

The paper presents an adoption of transition networks to recognizing and translating fragments of texts characteristic of e-mails written in Polish. An extension of XTND (XML Transition Networks Definition) – abbreviated as PTND – is introduced in order to provide a convenient way to describe these networks. A new tool for graphical representation, modification and validation of network descriptions is proposed.

Key Words

Machine Translation, Transition Networks, XTND, Drawing Tools

1. Introduction

POLENG is a rule-based Polish-to-English machine translation system, which has been developed since 1995. In the years 1995 – 2001 POLENG had a status of strictly academic project. Since 2002 the research has been directed towards commercial use. The first commercial partner of the project was the Allied Irish Bank (AIB) who envisaged the system as a communication tool between its centre in Ireland and its branches in Poland. One of the most urgent needs of AIB was the translation of e-mails.
Examination of an e-mail corpus delivered by AIB has shown that most e-mails contain texts that would not be translated correctly by a rule-based translation algorithm. This concerned mainly such parts of e-mails as greetings, farewells, adding attachments or expressions of gratitude or respect.
A few potential solutions to this problem have been considered. Example-based translation was rejected because of scarcity of bilingual e-mail corpora. Lexicon-based approach, i.e listing of all possible phrases characteristic of e-mails seemed to burden the system with heavy data of minimal use.
The regularity of these expressions made transition networks a convenient tool for their description and translation.

2. The origin of the idea

Transition networks (or transducers) have long been hoped to be a useful tool for translation. The reality has revised these hopes but the tool is still used for recognizing simple patterns prior to the translation process. SYSTRAN researchers report the use of translation networks for recognizing Hungarian temporal expressions in Hungarian-English translation [Senellart 2001a] and for translation of support files in the AUTODESK system [Senellart 2001b].
The approach presented here is different from the one suggested by SYSTRAN. We do not construct target (i.e. English) networks, and thus we do not make any alignment between networks of two languages. Our experiments on networks composed for both languages have shown that rules of alignment are of almost the same complexity as those of rule-based transfer.
We have decided that transition networks should generate equivalent English text in the process of recognition of a Polish pattern. In order to overcome well-known restrictions of transducers (e.g. discrepancies between orderings of words) we have augmented each network with a history stack, which stores the information about visited states, traversed transitions (and their pre-conditions) as well as executed actions.

3. PTND - POLENG Transition Network Definition

The format of our description for transition networks – that will be further on referred to as PTND (POLENG Transition Network Definition) – is based on XTND, the XML Transition Network Definition ([XTND, 2000]). The aim of the authors of XTND was to create a standard format for transition networks in a universally used language.

3.1. XTND – XML Transition Network Definition

The note [XTND, 2000] realizes two objectives: it gives a general and universally applicable definition of a transition network and provides a tool for the formal description of all elements of a network. This description is given in the XML language and forms a DTD (Document Type Definition). For example, the state is defined as an object with the following type:

```
state = object {
      name        = string;
      preconditions     = set of predicates;
      prelude     = ordered set of actions
      postconditions = set of predicates
      postlude    = ordered set of actions
};
```

A state is represented by its name, a set of *preconditions, i.e.* conditions that must be satisfied to enter the state, a set of actions that should be executed before entering the state, a set of *postconditions,* i.e. conditions that must be satisfied to exit the state and a set of actions that should be executed on exiting the state. The above definition as mirrored in the DTD for the XTND in the following way:

```
<!ELEMENT   state {properties?,
                   preconditions?,
                   prelude?,
                   postlude?,
                   postconditions?) >
<!ATTLIST state   id    ID          #REQUIRED
                  name  CDATA #REQUIRED >
```

The transition is defined as an object which consists of the following elements: *from, to, preconditions, actions*. The element definition of a transition in DTD is analogous to that of a state.

XTND suggests also a language for marking actions, called XEXPR, an example of which follows:

```
<set name="AA" value="AA Value"/>
<print newline="true"><get name="AA"/></print>
```

3.2. Applying XTND to PTND

In PTND the conditions are assigned to transitions only (and not to states). Actions are executed only during transitions. The notion of 'event' is not used in PTND.

3.3. Conditions in PTND

Transition condition is a "condition that guards the transition from one state to another" ([XTND, 2000]). In PTND a condition may have one of the following types:

- Empty
- Raw – string of letters
- Netref – reference to a subnetwork
- Code

The syntax of *Code* is the same as the syntax for 'condition' in the C language.

The need for using *Code* appears for *dictionary conditions*, i.e. conditions which have to be verified by looking up the dictionary of the POLENG system (see section 5. for Examples).

3.4. Actions

Action is an operation performed when a transition is traversed. In PTND actions are expressed with a sublanguage of C – limited to logical operations, comparisons and assignments. The code for actions is left verbatim (except for variables) when transition networks are compiled into a C++ code (see section 6).

3.5. Variables

The language used in conditions and actions allows for using variables. Two types of variables are used:

- local read-only variables, marked in the examples with a character '@'
- global read-write variables, marked in the examples with a character '$'.

3.6. References to stacked arguments

In order to be able to cope with variations of order of Polish expressions, each network is augmented with a history stack. The stack allows for referring to the values of local variables of previous transitions. For example, GEN[2] refers to the value of the local variable GEN that was set two transitions back.

4. The backtracking algorithm

A typical backtracking algorithm for transition networks goes back following the trace in order to find a state which still has some untraversed transitions exiting from it. The only information that has to be stored in such an algorithm is the list of visited states and the list – for each state – of untraversed transitions that exit from the state.

The backtracking algorithm that we use stores the information on transitions. If a transition is based on a network, then the success path is remembered in order for the algorithm to be able to look for another path in the backtracking process (if needed). If a transition is of a dictionary type and a dictionary entry is ambiguous, then the backtracking algorithm needs to know which interpretations of the dictionary entry have already been verified.

It is worth noting that in our approach backtracking is called not only upon failure but also in the reference to historical values of local variables.

5. The drawing tool

Experience has shown that even a universal standard of storing networks (like XML) raises difficulties in creating networks that are compatible both with linguistic data, and the formal description of the network. We have designed a drawing tool, called Gichon, which:

- allows for creation and modification of a network consistent with PTND
- reads a PTND description of a transition network and displays its graphical representation
- validates the description of a network against PTND
- stores a graphically designed (or modified) network in .xml file consistent with PTND
- checks the syntax of actions and conditions.

We have considered using some public domain software for drawing networks. One of the user-friendly public domain drawing tools is DIA. It was formerly designed for LINUX systems (see [Toga, 2001] for the description of the tool). Nowadays it is available also for Win32 systems (downloadable from htttp://hans.breuer.org/dia).

Eventually we have made the decision to design a new tool. The advantages of having a tool dedicated especially for our needs are following:

- Gichon, apart from drawing capabilities, supports the PTND formalism. It also means that the application can be easily re-designed in order to support XTND.
- Gichon is 100% Java code, which makes it platform-independent. The same application may be used both for LINUX and Win32.
- Gichon allows for checking the syntax of the Action language.

The next section presents examples of PTND networks as well as their graphical representations produced by Gichon. The application is downloadable from the POLENG web site: http://ceti.pl/~poleng.

3.6 Example

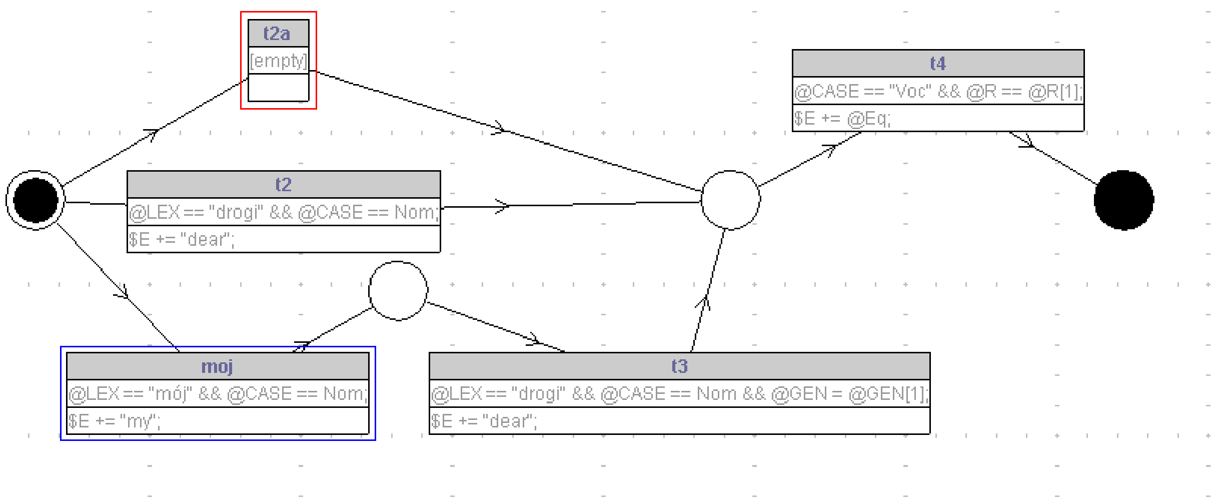
Below a part of description of a network that recognizes and translates a type of e-mail greetings, is presented:

```
<transition from="s1" id="t1" name="t1" to="s11">
      <conditions>
            <![CDATA[@LEX == "mój" && @CASE == Nom;]]>
      </conditions>
      <actions>
            <![CDATA[$E += "my";]]>
      </actions>
</transition>
<transition from="s11" id="t2" name="t2" to="s21">
      <conditions>
            <![CDATA[@LEX == "drogi" && @CASE == Nom && @GEN = @GEN[1];]]>
      </conditions>
      <actions>
            <![CDATA[$E += "dear";]]>
      </actions>
</transition>
<transition from="s21" id="t3" name="t3" to="s31">
      <conditions>
            <![CDATA[@CASE == "Voc" && @GEN == @GEN[1];]]>
      </conditions>
      <actions>
            <![CDATA[$E += @Eq;]]>
      </actions>
</transition>
```

The network recognizes exemplary headers: „Moja droga Julio" and "Mój drogi chłopcze" and produces „My dear Julia" and "My dear boy" respectively".
The first transition recognizes all nominative forms of the lexeme „mój" (i.e. singular, plural forms of all genders), the second transition recognizes nominative forms of the lexeme „drogi" and verifies consistency of the gender with the gender of the word recognized in the previous transition. The third transition recognizes vocative forms of Polish nouns and adds the equivalent found in the POLENG dictionary (@Eq) to the value of the global variable $E.
Picture 1. shows the representation of the above network produced by Gichon.

6. Using transition networks in the translation process

Similarly to SYSTRAN's approach, the POLENG mechanism of transition networks is used only for recognizing regular patterns prior to the translation process. The networks are applied to a tokenized text, i.e. a list of tokens, which represents the input text. The output of the recognition process is a list of tokens, based on the input list, with all segments of texts recognized by networks being marked appropriately. The recognized segments form units, which are described by the same data structures as 'regular' tokens. This means that the input and output list of recognition are of the same format. The output list of network recognition is transferred to the proper translation process. The translation procedures 'are unaware' of the fact whether any transition networks have been actually applied to the text. In other words, the network recognition stage is transparent in the whole translation process as far as data structures are concerned.


t2a
[empty]
t4
@CASE == "Voc" && @R == @R[1];
$E += @Eq;
t2
@LEX == "drogi" && @CASE == Nom;
$E += "dear";
moj
@LEX == "mój" && @CASE == Nom;
$E += "my";
t3
@LEX == "drogi" && @CASE == Nom && @GEN = @GEN[1];
$E += "dear";

Picture 1. Gichon's visualization of an exemplary TN

It would be ineffective to call all transition networks for any type of texts and at any position. Recognition according to a specific network takes place only when the POLENG system is in a definite *state*. The state comprises information about the domain of the text (e.g. *information technology*, *banking, science*), the type of the text (e.g. *e-mail, WWW page*), the current position in the text (e.g. *the beginning of the text*). For example, the network for recognizing greetings is checked only at the beginning of an e-mail text. Naturally, the networks themselves are allowed to change the state of the system.
Network recognition procedures are implemented in a C++ code. For convenience sake, the XML representations of networks are converted into pure C++ code. Transition conditions are converted into statements of conditions in *if* instructions. Actions are embraced by curly braces in order to create C++ statements. Network variables (attributes) inside conditions and actions are detected and converted into valid C expressions.

7. Conclusions

The research has led to the following conclusions:

- Transitions networks may still find their applications in describing some restricted phenomena of natural language.
- It has been a good idea to create the XTND specification. The document can be easily adopted for various implementations of transition networks.
- In order to deal with free-order languages transition networks must be augmented with stacks, which store the history of transitions.

We believe that the tool presented here may prove helpful in easy and convenient describing of quite complex patterns of natural language.

References

[Senellart 2001a], Senellart J., Dienes P. Varadi T., *New Generation Systran Translation System,* MT Summit VIII, Santiago de Compostela 2001
[Senellart 2001b], Senellart J., Plitt M. Bailly C, Cardoso F., *Resource Alignment and Implicit Transfer*, MT Summit VIII, Santiago de Compostela 2001
[XTND 2000] XTND - *XML Transition Network Definition, W3C Note 21 November 2000*, available at: http://www.w3.org/TR/xtnd/
[Toga 2001], Toga K. *DIA. Creating Charts and Diagrams*, available at: http://www.togaware.com/linuxbook/dia.pdf