Narzędzia do opisu i interpretacji skończonych sieci przejść w systemie POLENG

Tools for the Description of Finite Transition Networks in the POLENG System

Krzysztof Jassem, Jassem@amu.edu.pl

Tomasz Kowalski chiro@interia.pl

Uniwersytet im. Adama Mickiewicza, Wydzial Matematyki I Informatyki, Poznań

Streszczenie

W pracy przedstawiono zastosowanie sieci przejść do automatycznego tłumaczenia na język angielskich pewnych typów fragmentów polskich tekstów, które charakteryzują się regularną budową i wymagają specyficznych reguł translacji. Zaproponowano formalizm opisu sieci, oznaczony jako PTND, który opiera się na standardzie XTND (XML Transition Networks Definition) – uniwersalnym formalizmie reprezentacji sieci przejść. Opisano narzędzia do obsługi sieci przejść zgodnych z formalizmem PTND. Są to: interpreter graficzny, który umożliwia wizualizację sieci, generator kodu C, który konwertuje sieci do kodu języka C, dzięki czemu możliwe jest zastosowanie sieci w systemie POLENG, oraz produktor – narzędzie do testowania poprawności skonstruowanych sieci.

SUMMARY

The paper presents an adoption of transition networks to automatic translation of certain fragments of Polish that have characteristic structure and require specific rules for transfer. A formalism of network description, abbreviated as PTND, is advanced. PTND is based on XTND (XML Transition Networks Definition) – a universal formalism for the representation of transition networks. The paper describes computer tools designed to support the PTND formalism. These are: a graphical interpreter that allows for visualization of networks, a C generator that converts network descriptions into the C code, which makes networks applicable in the POLENG system, and the producer – a tool for testing and validating the networks.

~~1.~~1. Wstęp

Sformatowane: Punk

Algorytm tłumaczenia w systemie POLENG oparty jest na informacjach zaczerpniętych z dwujęzycznego leksykonu oraz zestawie

reguł analizy transferu i syntezy. W wyniku analizy składniowej uzyskiwana jest reprezentacja składniowa wyrażenia wejściowego w formie drzewa struktury frazowej. Reprezentacja ta przekształcana jest w odpowiadającą jej strukturę składniową wyrażenia w języku docelowym w oparciu o reguły transferu. Reguły syntezy stanowią podstawę do wygenerowania tekstu języka docelowego na podstawie struktury składniowej.

Okazuje się, że w autentycznych tekstach występuje wiele zjawisk, które z jednej strony wykazują bardzo regularny charakter, a z drugiej wymagają tłumaczenia w specyficzny dla siebie sposób. Przykładami mogą być liczby, daty, adresy czy personalia. Wydaje się celowe, by w tłumaczeniu tego typu wyrażeń stosować inny mechanizm niż stosowany dla ogólnych tekstów. Z jednej strony mechanizm taki mógłby opierać się na słabszym formalizmie reguł (ze względu na regularność opisywanych wyrażeń), z drugiej umożliwiałby zastosowanie wyróżnionych reguł transferu.

Do opisu wyrażeń o regularnej, choć specyficznej, budowie wybrano w systemie POLENG mechanizm skończonych sieci przejść. Okazuje się, że zastosowanie sieci przejść nie ogranicza się do wspomnianych typów wyrażeń. Wypracowany formalizm można stosować na przykład do tłumaczenia zwrotów charakterystycznych dla listów elektronicznych. Sposób wyrażania powitań, pożegnań, podziękowań, czy typowych sposobów przekazywania informacji dobrze poddaje się opisowi przy pomocy sieci przejść, przy czym poprawne ich tłumaczenie wymaga specjalnie skonstruowanych reguł.

W niniejszej pracy przedstawiony zostanie formalizm sieci przejść zastosowany w systemie POLENG, a przykładowe sieci opisywać będą niektóre wyrażenia charakterystyczne dla polskich listów elektronicznych.

2. Geneza pomysłu

W historii tłumaczenia automatycznego wielokrotnie sięgano po sieci przejść (czy też ich specyficzne realizacje zwane transduktorami – ang. *transducers*). Jednak wyniki z reguły satysfakcjonowały badaczy. W dzisiejszych czasach sieci przejść stosuje się tylko do rozpoznawania prostych wzorców przed samym procesem translacji. Na przykład w systemie SYSTRAN sieci przejść są stosowane w rozpoznawaniu określeń czasu w języku węgierskim [2], czy też w tłumaczeniu tekstów pomocy systemu AUTOSDESK [3].

Nasze podejście różni się od tego proponowanego w systemie SYSTRAN. Nie konstruujemy sieci docelowych i z tego powodu nie zajmujemy się odwzorowywaniem sieci z jednego języka na drugi. Wydaje się nam bowiem, że reguły odwzorowywania mogą mieć zbliżoną złożoność do ogólnych reguł transferu. Zdecydowaliśmy, że sieci przejść mają generować odpowiadające wyrażenia języka docelowego w trakcie procesu rozpoznawania wzorca. W celu przezwyciężenia problemu niezgodności

między porządkiem wyrazów wyposażyliśmy sieci w stos historii, w którym przechowywane są informacje o przebytych węzłach, łukach, warunkach i akcjach.

3. PTND - POLENG Transition Network Definition

Format opisu sieci przejść PTND (POLENG Transition Network Definition) opiera się na formalizmie XTND (XML Transition Network Definition, [4]). Zamierzeniem autorów XTND było stworzenie standardu opisu sieci w stosowanym uniwersalnie języku – XML.

3.1. XTND – XML Transition Network Definition

Formalizm XTND spełnia dwie funkcje: z jednej strony podaje uniwersalną definicję sieci przejść, z drugiej zapewnia spójny i uniwersalny sposób zapisu sieci przy pomocy języka XML. Specyfika (a zarazem uniwersalność) języka XML polega na tym, że pozwala on na istnienie dowolnie wielu swoich wersji. Autor swojej wersji języka charakteryzuje wymagania syntaktyczne przy pomocy tzw. DTD (Document Type Definition). DTD może być zrealizowany w formie osobnego pliku tekstowego lub zostać wkomponowany w dokument .xml (patrz np. [5] lub[6]).

XTND podaje wersję języka .xml opisaną w zewnętrznym pliku xtnd.dtd. Na przykład węzeł sieci definiowany jest jako obiekt następującego typu:

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

Węzeł reprezentowany jest przez swoją nazwę (name), zbiór warunków wejściowych, które muszą być spełnione, by można było wejść do węzła (preconditions), zbiór akcji wykonywanych w węźle (prelude), zbiór warunków opuszczenia węzła (postconditions) oraz zbiór akcji wykonywanych po opuszczeniu węzła (postlude). Definicja ta jest zrelizowana w XTND.DTD następująco:

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

Przejście definiowane jest jako obiekt o następujących elementach: *from, to, preconditions, actions*. Definicja elementu w XTND.DTD jest analogiczna do definicji węzła.

Ponadto w XTND proponowany jest język opisu akcji, zwany XEXPR, który jest próbą wyrażenia języka instrukcji w sposób deklaratywny. Oto próbka tego języka:

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

3.2. Elementy formalizmu XTND przejęte w PTND

W formalizmie PTND warunki dotyczą tylko przejść (a nie węzłów). Akcje wykonywane są wyłącznie podczas przejść. Nie jest wykorzystywane pojęcie 'zdarzenia'.

3.3. Warunki w PTND

Warunek przejścia strzeże przejścia z jednego stanu do drugiego (patrz: [4]). W formalizmie PTND warunek przejścia może być:

- Warunkiem pustym (Empty)
- Przejściem po zadanym wyrazie (Raw)
- Przejściem podsieci (Netref)
- Warunkiem podanym w określonym języku (Code)

  Składnia przejścia typu *Code* jest taka sama jak składnia warunku w języku C. Potrzeba takiego warunku pojawia się dla przejść *słownikowych*, czyli takich, które weryfikowane są poprzez słownik systemu POLENG.

3.4. Akcje

Akcja jest operacją, która zostaje wykonana podczas przechodzenia łuku. W formalizmie PTND akcje wyrażone są przy pomocy podzbioru języka C – ograniczonego do operacji logicznych, porównań i przypisań. Kod akcji jest wiernie kopiowany podczas kompilacji opisów sieci do języka C (patrz paragraf 6.).

3.5. Zmienne

Język opisu warunków i akcji dopuszcza jest stosowanie zmiennych. Wyróżnione są dwa typy zmiennych:

- zmienne lokalne do odczytu – ich nazwy poprzedzone są znakiem '@'
- zmienne globalne do odczytu i zapisu – ich nazwy poprzedzone są znakiem '$'

3.6. Referencje do argumentów z historii

Sieci wyposażone są w stos historii, co umożliwia w dowolnym momencie odwołanie się do wcześniejszych przejść. Dzięki temu możliwe staje się pokonanie trudności związanych z różnym szykiem wyrazów w odpowiadających sobie wyrażeniach. Stos historii pozwala również na weryfikację uzgodnień cech gramatycznych. Referencje wyrażane są przy

pomocy nawiasów kwadratowych, np. napis GEN[2] jest odwołaniem do wartości zmiennej lokalnej GEN ustawionej dwa przejścia wcześniej.

4. Algorytm nawracania

Typowy algorytm nawracania stosowany w sieciach przejść ma na celu dotarcie do węzła, z którego wychodzą nieprzebyte jeszcze łuki. Jedynym typem informacji wymaganym przez tego typu algorytm są listy odwiedzonych węzłów i przebytych przejść.

Algorytm stosowany w sieciach typu PTND wymaga przechowywania bardziej szczegółowej informacji na stosie historii takich jak: wartości zmiennych lokalnych i globalnych. Ponadto w przypadku przejść słownikowych dla haseł niejednoznacznych algorytm w procesie nawracania potrzebuje informacji, które interpretacje hasła zostały już zweryfikowane.

Warto zwrócić uwagę na fakt, że w proponowanym tutaj podejściu mechanizm nawracania może zostać wywołany w dowolnym momencie – nie tylko w przypadku porażki.

5. Interpreter graficzny

O ile zgodność opisu sieci z wymaganym formalizmem łatwo jest sprawdzić automatycznie, o tyle trudno jest zautomatyzować sam proces tworzenia sieci. Można jednak ten proces ułatwić poprzez skonstruowanie informatycznych narzędzi do rysowania sieci. Dla potrzeb formalizmu PTND stworzono narzędzie o następujących cechach funkcjonalnych:

- tworzenie i modyfikacja diagramów sieci zgodnych z PTND
- odczyt opisu PTND sieci i wyświetlenie jej graficznej reprezentacji w formie diagramu
- walidacja zgodności opisu z PTND
- zapamiętanie i przechowanie opisu sieci i wyglądu diagramu sieci
- sprawdzenie poprawności składniowej warunków i akcji

6. Generator

Graficzne przedstawienie sieci przejść, jako diagramu złożonego z umownych, połączonych ze sobą liniami, symboli rozmieszczonych na płaszczyźnie jest formatem najbardziej czytelnym dla człowieka.

O ile maksymalna czytelność dla człowieka jest punktem ciężkości w fazie tworzenia sieci, o tyle w momencie zastosowania sieci przejść w aplikacji komputerowej, np. w procesie tłumaczenia automatycznego, pożądana jest jak największa czytelność dla maszyny. Rozumiemy tutaj, że format danych jest tym bardziej czytelny dla maszyny, im mniej pracy należy włożyć w opracowanie algorytmu przetwarzania tych danych.

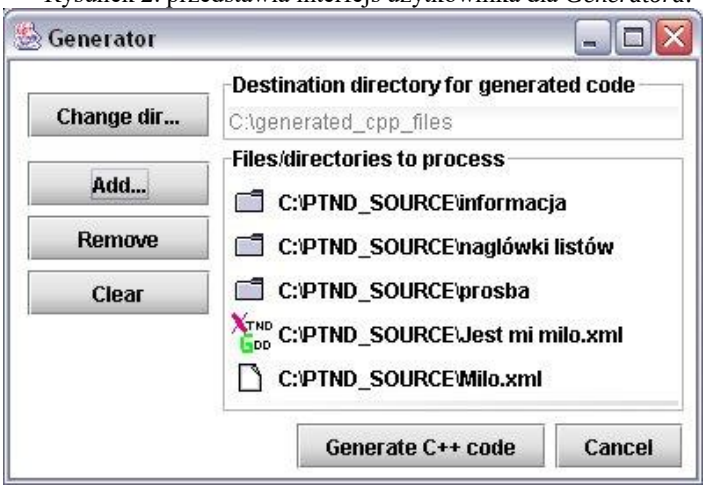
Celem *Generatora* jest skonwertowanie opisu sieci do formatu jak najbardziej czytelnego dla maszyny.

Z punktu widzenia jednostki przetwarzającej optymalna (najmniej złożona obliczeniowo) byłaby sytuacja, w której interpretacji podlegałby tylko ciąg wejściowy, a nie dodatkowo jeszcze sama sieć (jej opis). Wtedy każda sieć musiałaby stanowić samodzielny program! Nie jest to jednak pożądana sytuacja, gdyż wymaga by autor opisu sieci (specjalista z dziedziny opisywanej przez sieć, np. lingwista) był programistą.

Proponowane tutaj rozwiązanie polega na stworzeniu jednej ogólnej aplikacji do interpretacji ciągu wejściowego przez dowolną sieć zgodną z PTND. Aplikacja ta rozszerzana jest o biblioteki skompilowanych opisów poszczególnych sieci. Konwersją PTND na kod źródłowy składający się na owe biblioteki, zajmuje się *Generator*.

Dla każdej sieci, ogólnie dostępne prototypy stanów i przejść, rozszerzane są o informacje specyficzne dla sieci, czyli przede wszystkim o interpretację warunków i akcji dla każdego przejścia. Po kontroli składni makro-języka, w którym są one wyrażone, następuje jego „kompilacja" do C++. Dla każdego stanu i przejścia tworzony jest kod obiektów je reprezentujących. Końcową fazą procesu jest wygenerowanie „zaczepu" umożliwiającego załadowanie sieci do pamięci.

Rysunek 2. przedstawia interfejs użytkownika dla *Generatora*.



6. Produktor

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 Examples

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

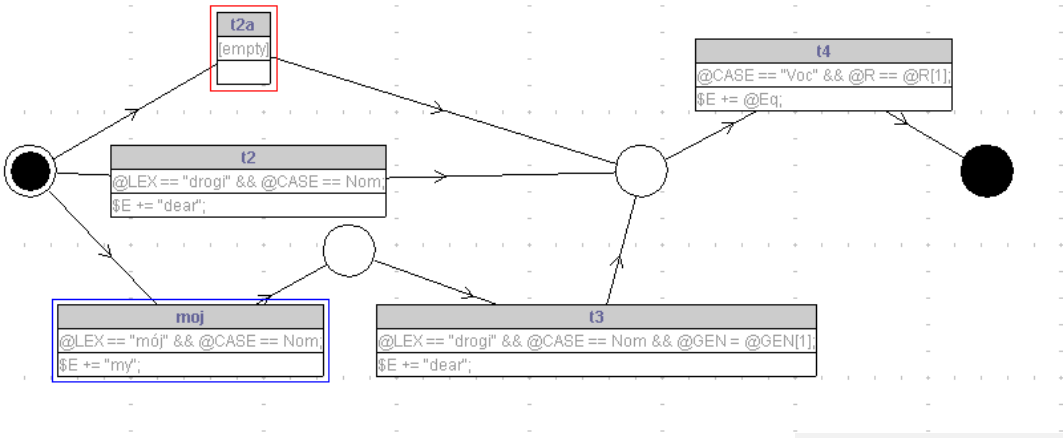
```
<transition from="s1" id="t1" name="t1" to="s11">
  <conditions>
        <![CDATA[@LEX == "mój" && @CASE == Nom;]]>
  </conditions>
  <actions>
        <![CDATA[$E += "my";]]>
  </actions>
</transition>
<transition from="s11" id="t2" name="t2" to="s21">
  <conditions>
        <![CDATA[@LEX == "drogi"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



Picture 1. Gichon visualization of an exemplary TN

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.

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 brought about 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

http://www/wdvl.co,/Authoring/HTML/Validation/DTD.html

http://www/w3schools.com/dtd

```
    [s1] -> (t8) -> [s2] -> (t12) -> [s4]
    { $WARTOSC=[] $POS=[] $R=[] $L=[] $P=[] $E=[very sorry]}
    [s1] -> (t5) -> [s3] -> (t7) -> [s4]
    { $WARTOSC=[] $POS=[] $R=[] $L=[] $P=[] $E=[very pleased]}
    [s1] -> (t5) -> [s3] -> (t6) -> [s4]
    { $WARTOSC=[] $POS=[] $R=[] $L=[] $P=[] $E=[very glad]}
    [s1] -> (t4) -> [s3] -> (t7) -> [s4]
    {   $WARTOSC=[]   $POS=[]   $R=[]   $L=[]   $P=[]   $E=[exceptionally
pleased]}
    [s1] -> (t4) -> [s3] -> (t6) -> [s4]
    { $WARTOSC=[] $POS=[] $R=[] $L=[] $P=[] $E=[exceptionally glad]}
    [s1] -> (t3) -> [s3] -> (t7) -> [s4]
    {   $WARTOSC=[]   $POS=[]   $R=[]   $L=[]   $P=[]   $E=[exceptionally
pleased]}
```