A Petri-Net-based Framework for Representing and Retrieving Conception in Case-based Document Writing

Chih-Hung Wu
Department of Information Management
Shu-Te Institute of Technology, Kaohsiung, Taiwan 824
Email: johnw@mail.stit.edu.tw

Abstract
Case-based document writing encounters inefficiency and incompleteness due to the use of explicit index-terms. We propose a Petri-Net framework which represents and retrieves the semantic concepts of cases for document writing. In our approach, cases of documents are represented as Petri Nets according to their semantic concepts. New documents can be made efficiently and completely by retrieving the conceptions of cases.

1 Introduction
Documentation is a painful and time-consuming process; especially, writing new documents for novices. In the process of writing, we usually refer to the publications or what we have written before. Sometimes, we may copy the whole document and make modifications on the materials which are directly related to the current topic. Clearly, this is a process of learning by experiences and may be solved by tools with artificial intelligence.

Case-Based Reasoning (CBR) collects happened problems and the effective solutions in a database, called a casebase, and applies these cases to new encountered problems. The application of cases in CBR is as follows.

• Collecting and storing cases into the casebase.
• Indexing the cases by some specific keywords or values appearing in cases.
• Retrieving cases for a new problem by searching the casebase using available indices.
• Adapting the contents of the retrieved cases to solve the new problem.

Intuitively, applying CBR techniques to document writing is a viable solution. Unfortunately, applying CBR to document writing by direct keywords or values encounters the following problems; restricting cases to be small and introducing inefficiency to the retrieval of cases.

• It is difficult to define index keys using direct keywords or values in a single case. In a single sentence, for example, there may be many keywords can be indexed. However, each indexed key may emphasize on different topics.
• If we take a complete document as a case, there would be thousands of sentences being the sub-cases. Among these sub-cases, many of them represent similar meanings but may have totally different indexed keys. Searching available cases using these indexed keys may be inefficient.
• Adaptation of cases is difficult when no cases is directly available. It is possible that two mergable cases are indexed by different keywords but can not be adapted for the new problem. For the similar reasons, it is very difficult to define an effective similarity function for all cases.
• It is very difficult to verify the completeness and correctness and consistency of the retained cases.

There need a lot of human efforts in selecting and adapting cases in such direct application of CBR to document writing. Therefore, we might conclude that when applying CBR to complex problems with a large number of cases to be processed, there must be a mechanism for organizing and analyzing the main features, or the conception, of the encountered problem. Retrieving of cases is based on the analyzed conception, not on the direct values or keywords, of the problem. Here we apply the Enhanced High-level Petri Net (EHLPN) to the representation and extraction of conceptions of writing problems in CBR.

Previous Work
CBR has received great successes in many fields such as planning, interior and architecture design, argumentation and legal reasoning, decision making, etc., refer to [2, 6, 1] for more discussion. There has been some researches working on computer-aided text-writing/generation [7, 9, 4, 5]. However, few applies CBR to document writing [3].

2 Conception of CBR for Document Writing
Below we give a brief discussion about how CBR works for document writing. For simplicity, we
only show how CBR takes care of writing sentences. Suppose that the following sentence is from an article describing a family.

"John is a 35-year-old father of two children."

A case for the above sentence may look like as follows.

<table>
<thead>
<tr>
<th>case ID</th>
<th>1-f-345</th>
</tr>
</thead>
<tbody>
<tr>
<td>TYPE</td>
<td>of</td>
</tr>
<tr>
<td>RELATION</td>
<td>is_father</td>
</tr>
<tr>
<td>INDEX</td>
<td>year-old, father, children</td>
</tr>
<tr>
<td>SOLUTION</td>
<td>John is a 35-year-old father of two children.</td>
</tr>
<tr>
<td>REF. ID</td>
<td>03-05-12, family.db</td>
</tr>
</tbody>
</table>

Note that the indices, "year-old", "father", and "children", are given by the users or an automated classifier. Using these indices, one can re-use the sentence if he/she wants to describe "Charly is 28 years old and he has 2 children". If we have two more cases as follows, there will be many combinations for constructing the sentence, "Charly is father of 3 happy children", but none for "Mary, a 25-year-old lady, is John’s mother".

<table>
<thead>
<tr>
<th>case ID</th>
<th>1-d-079</th>
</tr>
</thead>
<tbody>
<tr>
<td>TYPE</td>
<td>of</td>
</tr>
<tr>
<td>RELATION</td>
<td>is_father</td>
</tr>
<tr>
<td>INDEX</td>
<td>happy, father, children</td>
</tr>
<tr>
<td>SOLUTION</td>
<td>Mark is a happy father of one child.</td>
</tr>
<tr>
<td>REF. ID</td>
<td>02-11-03, family.db</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>case ID</th>
<th>m-x-391</th>
</tr>
</thead>
<tbody>
<tr>
<td>TYPE</td>
<td>of</td>
</tr>
<tr>
<td>RELATION</td>
<td>is_father</td>
</tr>
<tr>
<td>INDEX</td>
<td>father</td>
</tr>
<tr>
<td>SOLUTION</td>
<td>Joseph is father of John.</td>
</tr>
<tr>
<td>REF. ID</td>
<td>01-05-15, family.db</td>
</tr>
</tbody>
</table>

These cases seem to be different, but similar in describing a concept, the parent_and_child relation.

3 Conception of Cases

Direct access to the casebase using explicit index-keys will cause inefficiency and may lost some available cases. In case-based document writing, the retrieval of cases from a large casebase (image that how many sentences are there in an article) means a mechanism for retrieving the concepts of cases. For the length of the paper, the definitions in this section are informally given. We define that the fundamental relations in a sentence as a primitive relation which are represented in first-order logic. For example, the primitive relations in the three cases are as follows.

- f-j-345: father(john,X), father(john,Y), age(john,35), has_kids(john,2).
- f-d-679: father(mark,X), mood(mark,happy), has_kids(mark,1).
- m-x-391: father(joseph,John).

We define the most specific common primitive, \(\rho\), of a group of primitive relations to be a literal which can be instantiated to be any one of the primitive relation in the group. For example, \(\rho\) of father(john, joe), father(john,mary), father(john,helen) is father(john,X) and \(\rho\) of father(john,X) and father(joseph,john) is father(X,Y).

A conception of a group of cases to be a semantic network which hierarchically represents the common semantics of two or more sentences. The inner nodes in a conception are primitive relations or \(\rho\). For example, the conception of the cases, f-j-345, f-d-679, and m-x-391, is depicted in Figure 1. Conceptions not only represent the generally concepts of the retained sentences but also the inter-relationship among sentences. Note that, there may be more than one conceptions each of which may represent different concepts in a casebase.

4 Representing/Retrieving Concept by Petri Nets

Enhanced High-Level Petri Nets (EHLPNs) [10] expand the expressive power of traditional Petri nets [8] for conditional systems containing variables and negative information. Suppose \(U\) is a finite set of constants. \(U^k\) denotes the set of all \(k\)-tuple \(\langle u_1, u_2, \ldots, u_k \rangle\), where each \(u_i \in U\). Also, suppose \(V\) is a finite set of variables ranging over \(U\). \(V^k\) denotes the set of all \(k\)-tuple \(\langle v_1, v_2, \ldots, v_k \rangle\), where each \(v_i \in V\). Let \(W = U \cup V\), such that \(W^k\) denotes the set of all \(k\)-tuple \(\langle w_1, w_2, \ldots, w_k \rangle\), where each \(w_i \in W\). Each element in \(W^k\) has the same arity, \(k\). The arity of \(X\) is denoted as \(a(X)\).

Definition. An Enhanced High-level Petri Net is an 8-tuple \(N = (P, T, F, L, U, V, W, \mu)\), \(F = E_3 \cup F_r\), \(L = L_0 \cup L_r\), such that

1. \(P, T, \) and \(F\) form a Petri Net structure.
2. \(U\) and \(V\) are the sets of constants and variables, respectively, appearing in the underlying system, and \(W = U \cup V\).
3. \( L_0 : F_0 \rightarrow \bigcup_{i=0}^{n} W^i \) and \( L_r : F_r \rightarrow \bigcup_{i=0}^{n} W^i \), where \( n \) is the maximum arity of all \( p \in P \).

4. \( \mu : P \rightarrow V, \mu = \mu_L \cup \mu_r, \mu_L \) and \( \mu_r \) describe the distributions of tokens, which represent the occurrence of positive and negative events, respectively.

5. \( \forall p \in P \) and \( \forall x, y \in \bigcup_{(a,b) \in P} (L((a,b))) \), we require \( \alpha(x) = \alpha(y) \).

The elements of \( P, T, F_0, F_r \), and \( L \) are called places, implications, excitant-arts, inhibitor-arts, and arc labels, respectively, of \( N \). For a transition \( t \in T \), the arguments of \( t \) are defined to be the distinct variables associated with \( t \). Let \( V_1, V_2, \ldots, V_{\alpha(t)} \) be the variables associated with \( t \). A place \( p \in P \) connected with \( t \) presents a positive or negative relationship between \( p \) and \( t \). The relationship is strictly specified by the label \( l \in L \) of the arc \( f \in F \) connecting \( p \) and \( t \). If \( l \in L_0 \) and \( f \in F_0 \), \( p \) is called a positive place of \( t \) and there might exist a substitution \( \Theta \) of variables such that both \( p((l)\Theta) \) and \( t(V_1,\ldots,V_{\alpha(t)}\Theta) \) are true. If \( l \in L_r \) and \( f \in F_r \), \( p \) is called a negative place of \( t \) and there might exist a substitution \( \Theta' \) of variables such that both \( \neg p((l)\Theta') \) and \( t(V_1,\ldots,V_{\alpha(t)}\Theta') \) are true.

Let a set of \( k \)-tuples \( C(x) = \bigcup_{(a, \alpha)} x \in P \cup T \), be the color set of \( x \). Every element in \( C(x) \) is called a color of \( x \) as follows.

**Definition.** For a \( k \)-arity place \( p \in P \) connected with an \( m \)-arity transition \( t \in T \) by an arc \( f \in F \) labeled as \( l = (W_1,\ldots,W_k) \), a color \( c = (a_1,a_2,\ldots,a_k) \) is a ground substitution for \( p(W_1,\ldots,W_k) \), where \( c \) is a **blue**-color if it signifies that the predicate \( p(a_1,a_2,\ldots,a_k) \) is evaluated to be true, and \( c \) is a **red**-color if it signifies that the predicate \( \neg p(a_1,a_2,\ldots,a_k) \) is evaluated to be true. A color \( \theta = (a_1,a_2,\ldots,a_m) \) of \( t \) denotes a substitution for all the variables specified in \( t \), signifying that there is a set of colors, each of which is from one of \( t \)'s inplaces, evaluated to be true according to \( \theta \).

The enabling and firing of transitions in EHLPN is similar to traditional Petri Nets, except that transitions are enabled in EHLPN if all the positive inplaces have blue-colors and all the negative inplaces have red-colors and the variable binding among these colors are consistent. The change of marking in EHLPN after firing transition is the same as traditional Petri Nets, i.e., inplaces of the fired transitions remove associated colors and outplaces receive associated colors with proper variable bindings. Refer to [10], for more detailed definitions.

**Modeling Conception in EHLPNs**

Given a casebase, EHLPN can directly represent conceptions as follows.

---

Figure 2: Representing conceptions using EHLPN

1. For a literal, \( p(v_1, v_2, \ldots, v_n) \), representing a \( \rho \), create a place \( p \) connected with an output arc labeled as \( (v_1, v_2, \ldots, v_n) \).

2. If two or more \( \rho \)'s form a sentence, create a transition \( t \) for the semantics of the sentence, where all \( \rho \)'s are connected to \( t \) with their output arcs.

3. For the connector, \( y \), of a sentence, create a transition, \( t_i \), with inplaces, \( p_y \) and \( y \). The outplace, \( p_y \), of \( t_i \) represents the semantics of the sentence, whose arc is labeled by the same as the arguments of \( t \).

4. For each \( t_i \), create a transition, \( t_o \), without outplaces. The inplaces of \( t_o \) are \( o_i \) and the case sentence.

5. For all each instance, \( p(m_1, \ldots, m_n) \), of \( \rho \), create blue/red-colors for \( (m_1, \ldots, m_n) \) and put them in the place \( p \). Note that, positive expressions are places connected with excitant-arts, and negative expressions by inhibitor-arts.

6. Semantic relation can be directly represented in this EHLPN. For completeness, each inplace of such transitions is connected with an output arc from the transitions.

Figure 2 presents the corresponding EHLPN structure for the child-and-parent problem mentioned previously.

**Retrieving Conceptions**

Given an EHLPN \( \mathcal{N} \) corresponding to the conceptions of a casebase, the process of retrieving conceptions of cases is as follows. For the length of the paper, we take the parent-and-child problem as an example to show how it works. Suppose that we are
writing a sentence expressing that “Candy is 45 years old” and “Candy has one child”. Let the primitive relations in this sentence be \( S = \{ \text{mother(candy,X)}, \text{age(candy,45)}, \text{has.kids(candy,1)} \} \) and the the primitive relations in the casebase be \( I \). We also assume that the relationships parent(X,Y)←father(Y,X) and parent(X,Y)←mother(Y,X) have already been described in \( N \).

- In Figure 2, put all primitive relations as blue-/red-colors to associated places. Then, the places, father, mother, and age, have some blue-colors.

- Firing transitions which can be enabled by the colors from \( S \). The transition, \( t_1 \), is enabled and fires. The firing of \( t_1 \) produces a blue-color \( \langle \text{candy}, X \rangle \) to the place parent.

- Firing transitions which are enabled by the colors from \( I \). The transition, \( t_2 \), is enabled and fires. The firing of \( t_2 \) produces blue-colors \( \langle \text{john}, X \rangle, \langle \text{john}, Y \rangle, \langle \text{mark}, Z \rangle \) to the place parent. Also, the place age has a blue-color \( \langle \text{john}, 35 \rangle \) and \( \langle \text{candy}, 45 \rangle \).

- Observe the places which collect the colors produced from \( S \) and \( I \). In Figure 2, such places are parent.

- Firing transitions which are enabled by \( I \) again, until cases can be reached. Here, transitions \( t_3 \), \( t_4 \) and \( t_4 \) are enabled and fire in sequence and the place case receives a blue-color \( \langle f - j - 345 \rangle \).

- Adjusting the cases according to the relation of father, mother, and parent. According to \( \rho \) of age(X) and the interaction between \( t_1 \) and \( t_2 \) through the place parent, we modify “f-j-345” by changing “John” to “Candy” and “father” to “mother” and “35” to “45” and “2” to “1”, and obtain a sentence “Candy is a 45-year-old mother of 1 children”. Of course, we have modify the obtained sentence according to grammar and change “children” to “child”.

5 Discussion and Conclusion

The use of conceptions makes the applicable space of cases dramatically broader. In this way, cases of documents can be applied to problems which are not directly related to existing cases. For instance, the conceptions and cases about human family can be applied to stories about “the motherhood of wild animals”. The techniques we have presented in this paper are applied to sentence-level document writing. However, they can be easily applied to higher levels, such as writing the organization for a complete document or a chapter or a paragraph. Since verification and validation of knowledge can be easily and completely done using EHLNP, The verification/validation of cases can also be achieved in the same way.

References


