An XML-based Petri Net to Rules Transformation Software Tool

Kiatnarong Tongprasert, Suphamit Chittayasothorn

School of Computer Engineering, Faculty of Engineering

King Mongkut’s Institute of Technology, Ladkrabang, Bangkok 10520

ABSTRACT

This research work presents a software tool which is used to generate ruleml rules from a Petri Net structure in the pnml format. A transformation algorithm is presented and illustrated examples are shown. The algorithm employs a depth-first search technique. The generated result is comparable to the result which was presented by other research works.

Key Words: Petri Nets, Rules, pnml, ruleml

1. INTRODUCTION

Knowledge-based expert systems have been around with numerous applications for decades. With the Internet and related technology available, such systems are now available on the web. Inference engines which are developed in Java such as jDrew [1] can read and infer rules in XML format. A well-accepted format is the ruleml [2]. Rule is in fact one of the most widely used knowledge representation in knowledge-based systems nowadays. The development of a rule-based system seems to be simple and straightforward. However, the hard part is to check the consistency of the rules especially after the addition of new rules and the removal of old ones. To overcome this problem, a modeling technique which can be used to model rules in the knowledge base for consistency checking is required. Petri Net [3], [4] has been adopted for the task by [5] and [6]. Some research works such as [7] and [8] even use Petri Net as their knowledge representation. Also, pnml [9], a version of Petri Net is now available in the XML format. This research paper investigates the appropriateness of using Petri Nets in pnml format to model rule-based systems in ruleml format. A software tool that accepts pnml-based Petri Nets and transforms them into ruleml-based rules for the construction of a consistent knowledge base is developed.

2. PETRI NET

Petri Net is a dynamic simulation tool which is widely used to simulate dynamic behaviors of a system. It was invented by Carl Adam Petri in 1962 [3]. Its popular applications include the simulation of concurrent operations, synchronization, and parallel systems. Petri Net gains its popularity because of its mathematical background and its graphical presentation. Major components of Petri Net include places, transitions, and tokens.

Places are symbolized by circles or ellipses can be further classified into input and output places. An input place represents a precondition of an event and an output place represents a postcondition of an event. Table 1 summarizes the interpretation of places and transitions.

<table>
<thead>
<tr>
<th>Input Place</th>
<th>Transition</th>
<th>Output Places</th>
</tr>
</thead>
<tbody>
<tr>
<td>Preconditions</td>
<td>Event</td>
<td>Post conditions</td>
</tr>
<tr>
<td>Input data</td>
<td>Computation step</td>
<td>Output data</td>
</tr>
<tr>
<td>Input signals</td>
<td>Signal processor</td>
<td>Output signals</td>
</tr>
<tr>
<td>Resource needed</td>
<td>Task or job</td>
<td>Resources released</td>
</tr>
<tr>
<td>Conditions</td>
<td>Clause in logic</td>
<td>Conclusion</td>
</tr>
<tr>
<td>Buffers</td>
<td>Processor</td>
<td>Buffers</td>
</tr>
</tbody>
</table>

PNML (Petri Net markup Language) is an XML based language which is used to exchange Petri Nets between applications. It is now an industrial standard (ISO/IEC 15909-2:2009).

After a Petri Net is constructed and approved by users, the next step is to transform it into rules so that the rules can be implemented in a knowledge-based system. A transformation of a Petri Net to rules can be done automatically. A transformation algorithm is presented in the next section.

3. A PETRI NET TO RULES TRANSFORMATION ALGORITHM

The transformation of a Petri Net into Rules is actually the traversal of the graph to find all nodes that can be reached from a given node. There are two kinds of node in the Petri Net graph namely the place and the transition. The begin point and the end point of each traversal are places. Relationships between places and transitions can be represented by an incident matrix. If an arc is output from a place to a transition, the corresponding matrix entry is -1. If the direction of an arc is from a transition to a place, the matrix entry is 1. Figure 1 shows a Petri Net and Figure 2 shows its corresponding incident matrix.
An algorithm which transforms a Petri net into rules is shown as follows:

1. Choose place $p_i$ to begin the construction of a tree $t_{ee}$
2. Add place $p_i$ into the tree $t_{ee}$
3. For each transition $t_m$ at place $p_i$ which has a negative value:
   3.1. If transition $t_m$ is not in the tree $t_{ee}$, add the transition $t_m$ into the tree $t_{ee}$
   3.2. For each place $p_j$ which participates in transition $t_m$
      3.2.1. If place $p_j$ does not terminate and is not in the tree $t_{ee}$, add place $p_j$ into the tree $t_{ee}$
      3.2.2. If the Incident matrix of the place $p_j$ does not have a positive value for all transition, terminate the place $p_j$ and exit from the loop.
      3.2.3. If the Incident matrix of the place $p_j$ in the transition $t_m$ has a positive value,
         3.2.3.1. If the place $p_j$ does not terminate is not in the tree $t_{ee}$, add the place $p_j$ into the tree $t_{ee}$ and go to 2
4. Find a place which is not in the tree $t_{ee}$, if found then construct a tree $t_{ee}$, and go to 2

As a demonstration we refer to the following medical diagnosis Petri net from [8] as show in Figure 3 and apply the above algorithm to it. An incident matrix show in Figure 4. A part of the result in ruleml is shown in Appendix A.

A rule is generated for each path of the above tree show in Figure 5. The three following rules are among those which are generated:

R1: $P_2 \land P_7 \land P_{13} \Rightarrow P_{22}$
P2: Male Signal
P7: No Fever Signal
P13: No Right Lower Quadrant (RLQ) Pain
Then $P_{22}$: Discharge

R2: $P_2 \land P_7 \land P_{12} \land P_{25} \Rightarrow P_{21}$
P2: Male Signal
P7: No Fever Signal
P12: Right Lower Quadrant (RLQ) Pain
P25: Positive CAT Scan Signal
Then $P_{21}$: Emergency Appendectomy

R3: $P_2 \land P_7 \land P_{12} \land P_{26} \land P_{19} \Rightarrow P_{23}$
P2: Male Signal
P7: No Fever Signal
P12: Right Lower Quadrant (RLQ) Pain
P26: Negative CAT Scan Signal
P19: Elevated White Blood Cell (WBC)
Then $P_{23}$: Admit

4. CONCLUSION

This research paper investigates the appropriateness of using Petri Nets in pnml format to model rule-based systems in ruleml format. A software tool that accepts pnml-based Petri Nets and transforms them into ruleml-based rules for the construction of a consistent knowledge base is developed.
Figure 3. The medical diagnosis Petri net from [8]

Figure 4. The corresponding incident matrix of the Petri net from Figure 3
5. REFERENCES


Appendix A: The Generated Rules in RuleML format

```xml
<RuleML>
  <Assert>
    <Implies>
      <head>
        <Atom>
          <op>
            <Rel>t24: Female with No Fever and No RLQ Pain Normal WBC Determination</Rel>
          </op>
          <Var>p22: Discharge</Var>
        </Atom>
      </head>
      <body>
        <And>
          <Atom>
            <op>
              <Rel>t2: Female Sex Identification</Rel>
            </op>
            <Var>p3: Female Signal</Var>
            <Var>p1: Idle Screenerp1</Var>
          </Atom>
          <Atom>
            <op>
              <Rel>t6: Female No Fever Determination</Rel>
            </op>
            <Var>p5: Female Patient</Var>
            <Var>p7: No Fever Signal</Var>
          </Atom>
          <Atom>
            <op>
              <Rel>t14: Female with No Fever and No RLQ Pain Determination</Rel>
            </op>
            <Var>p18: Female with No Fever and No RLQ Pain</Var>
            <Var>p20: Normal White Blood Cell (WBC)</Var>
          </Atom>
        </And>
      </body>
    </Implies>
  </Assert>
</RuleML>
```