Petri Net Reduction Using Association Rules Technique

Ass.Proof Dr.Ahmed Tariq Sadiq* Akbas Ezedeen*

ABSTRACT

There are several techniques to reduce Petri nets which play important roles in wide application areas. This paper presents a new approach to reduce the Petri nets. The suggested approach depends on converting the Petri nets to logic programs, then process the production rules of logic programs using association rules technique with Apriori algorithm to extract the logical which allows to reduce the logic relations of these production rules program. At last, the Petri nets are reconstruct from reduced logic programs to produce reduced Petri nets. The experimental results show that the suggested approach is a good, efficient and logical technique to reduce the Petri nets.

* Computer Sciences Department/ University of Technology/ Baghdad, Iraq

1-Introduction

A major weakness of Petri nets is the complexity problem. Thus, it is very important to develop methods of transformations which allow hierarchical or stepwise reductions and preserve the system properties to be analyzed. The main contribution is the principle of generalization of the reductions. Indeed the reduction theory is nothing but a heuristic method and then [11]:

- it provides only sufficient conditions to the simplification problem,
- it will always be improvable by more sophisticated rules.

The theory of Petri net reductions allows a given net to be replaced by a reduced net that maintains certain properties of the original net. The goal is to produce a Petri net which generates a significantly smaller reachability graph. Individual reduction steps can be defined as the replacement of a sub-graph of a given Petri net with a new graph. The conditions under which a reduction step may be applied are typically defined by the presence of a sub-graph with certain structural properties. Individual reduction steps tend to be small and inexpensive to perform. A sequence of reduction steps is applied iteratively to reduce the structure of the original Petri net, until no more reductions are applicable [4]. In this paper the suggested approach presents a new technique to reduce Petri nets by converting these to logic programs and association rules using Apriori algorithm to extract the relations of production rules.

2- Petri nets

Petri net is a graphical and mathematical modeling tool for describing, studying and analyzing the flow of information and control in systems. It has a great modeling power to represent different kinds of systems.

A Petri net [3, 12] is defined as the 5-tuple N=(P, T, E, M0, P and T are two disjoint sets of vertices called places and W). transitions, respectively. E is the set of arcs consisting of directed edges from a place to a transition or from a transition to a place. M0 is a function from P to the set of nonnegative integers and denotes the initial token distribution, called the initial marking. W is a function of E to the set weight or multiplicity assigned to arc e.

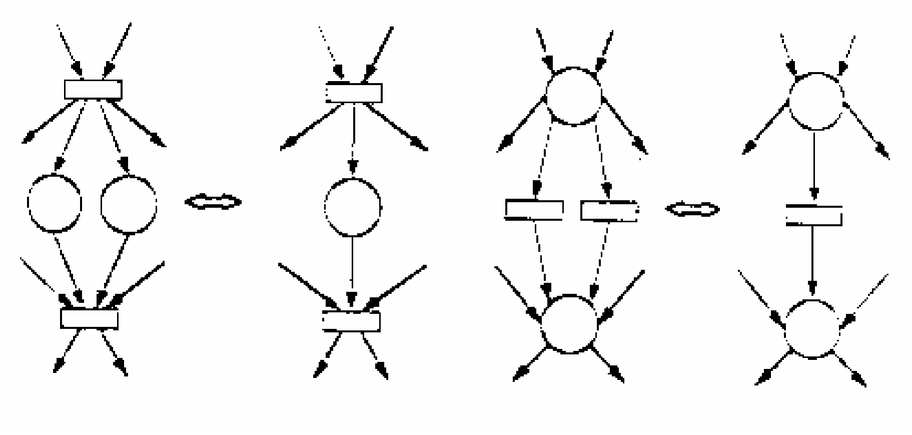
A transition is said to be enabled or firable if each of its input places has at least as many tokens as the weight of respective incoming arc to the transition. A firing of an enabled transition t removes w(p,t) token from each input place p of t and adds w(t,p) tokens to each output place p of t, where w(t,p) is the weight of the arc from t to p [13].

The dynamic behavior of systems modeled by Petri nets can be described by differential equations. The incidence matrix for a Petri net with n transitions and m places, A= [aij] is an n×m matrix of integers. aij is the weight of the arc from transition i to place j minus the weight of the arc from place j to transition i.

An n-vector of integers, x is called a T-invariant which satisfies the equation: $A^t.x = 0$ (where A^t is the transition of the incidence matrix A and the transpose is the operation of interchanging rows and columns of the matrix A) [2].

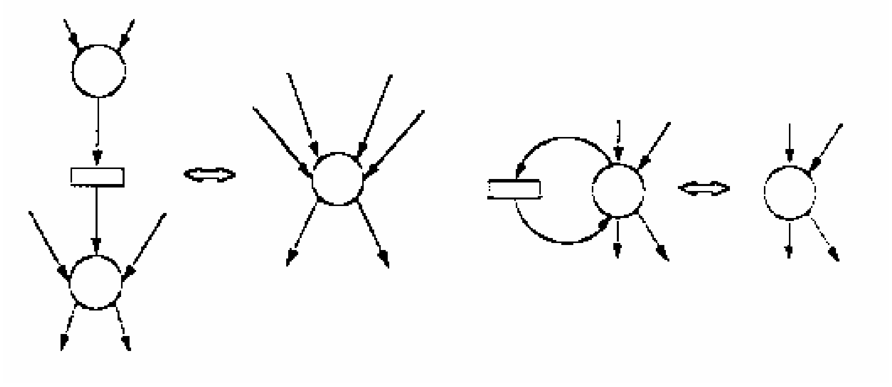
3- Reduction in Petri net

To facilitate the analysis of a large system, we often reduce the system model to a simpler one, while preserving the system properties to be analyzed. Techniques to transform an abstracted model into a more refined model in a hierarchical manner can be used for synthesis [13]. There exist many transformation techniques for Petri nets [13]. Some of these transformation techniques are depicted in Fig-1.



(a) Fusion of Parallel Transitions (FPT)

(b) Fusion of Parallel Places (FPP)



(c) Fusion of series Places (FSP) (d) Elimination of Self-loop Transition (EST)

Fig -1-
Petri Nets Reduction Transformation Types

4- Association Rules Concept

4-1 Association Rule Algorithms Definition

An association rule is a rule, which implies certain association relationships among a set of objects (such as those which occur together or one which implies the other), in a database. Given a set of transaction, where each transaction is a set of literal(called items), an association rule is an expression of the form XY, where X and Y are sets of items. The intuitive meaning of such a rule is that transactions of the database, which contain X, tend to contain Y [7]. Association rules identify relationships between attributes and items in database such as the presence or absence of one pattern implies the presence or absence of another pattern. Mining of such rules is one of the most popular pattern discovery methods in knowledge data discovery,

an association rule is an expression $X \Rightarrow Y$ where X={x1, x2… xn} and Y={y1, y2… yn} are set of items with left hand side (LHS) and right hand side (RHS). The meaning of such rules is quite intuitive: given database D of transaction T where each transaction $T \in D$ is a set of items $X \Rightarrow Y$ which expresses that whenever a transaction T contains X ,the T probably contains Y. Also the probability of rule strength is defined as the percentage of transactions containing Y in addition to X. The prevalence for rule is the percentage of transactions that hold all the items in the union. If prevalence is low, it implies that there is no overwhelming evidence that items in $X \cup Y$ occur together.

The rule $X \Rightarrow Y$ has support S in D if the fractions of the transactions in D contain $(X \cup Y)$ [9].

The problem of mining association rules is to generate all association rules that have certain user- specified minimum support (called min-sup) and confidence (called min-conf) [10]. The important measures for association rules, support (S) and confidence (C) can be defined as: The support (S) of an association rule is the ratio (in percent) of the records that contain $(X \cup Y)$ to the total number of records in database. Therefore, if we say that the support of a rule is 5% then it means that 5% of the total records contains $(X \cup Y)$. Support is the statistical significance of a rule [10].

Support $(X \Rightarrow Y) = P(X \cup Y)$

Support $(X \Rightarrow Y)$ = frequent (X) / total number of records in database.

For a given number of records, confidence (C) is the ratio (in percent) of the numbers of records that contain $(X \cup Y)$, to the number of records that contain X. thus, if we say that a rule has a confidence of 5% it means that 85% of the records containing X also contain Y. The confidence of a rule indicates the degree of correlation in the database between X and Y.

Confidence $(X \Rightarrow Y) = P(Y \cup X)$

Confidence $(X \Rightarrow Y) =$ frequent $(X \cup Y)$ / frequent (X)

Confidence is also a measure of rules strength. Mining of database consists of finding all rules that meet the user-specified threshold support and confidence.

4-2 Apriori Algorithm

Apriori is an influential algorithm for frequent itemsets for association rules which was introduced by Agrawal [8]. The name of the algorithm is based on the fact that the algorithm uses prior knowledge of frequent itemset properties. Apriori employs an iterative approach known as a level_wise search, where k-itemsets are used to explore (k+1)_itemsets. First, the set of frequent 1_itemsets is found. This set is denoted L_1. L_1 is used to find L_2, the set of frequent 2-itemsets, which is used to find L_3, and so on, until no more frequent k_itemsets can be found. The finding of each L_k requires one full scan of the database [4].

To improve the efficiency of the level-wise generation of frequent itemsets, an important property called Apriori property, is used to reduce the search space. *Apriori property* is based on this concept : If an itemset *I* does not satisfy the minimum support threshold (minsup), then *I* is not frequent, that is, $P(I) <$ min-sup. If an item *A* is added to the itemset *I* then the resulting itemset (i.e. $I \cup A$) cannot occur more frequently than *I*. Therefore, $I \cup A$ is not frequent either, that is, $P(I \cup A) <$min-sup, this property belongs to a special category of properties called anti-monotone in the sense that if a set cannot pass a test, all of its supersets will fail the same test as well. It is called anti-monotone because the property is monotonic in the context of failing a test. In general to find L_k two step process consisting of *join* and *prune* actions is used :

1- The Join Step: to find L_k is generated by joining L_{k-1} with itself. This set of candidates is denoted C_k, let *l1* and *l2* be itemsETS in L_{K-1}. The notation *li*[j] refers to jth item in *li* (*l1*[k-2] refers to the second to the last item in *l1*). Apriori assumes that items with a transaction or itemset are sorted in lexicographic order. The join, L_{k-1} join L_{k-1}, is performed, where members of L_{k-1} are joinable if their first (k-2)_items are in common ,that is members *l1* and *l2* of L_{k-1} are joined if (*l1*[1] = *l2*[1]) ^ (*l1*[2] = *l2*[2]) ^ …^ (*l1*[k-2] = *l2*[k-2]) ^ (*l1*[k-1] = *l2*[k-1]). The condition (*l1*[k-1] = *l2*[k-1]) simply ensures that no duplicates are generated. The result itemset is formed by joining *l1* and *l2* is *l1*[1] *l1*[2] …*l1*[k-1] = *l2*[k-1].

2- The Prune Step : C_k is a superset of L_k, that is, its members may or may not be frequent, but all of the frequent k_itemsets are included in C_k. A scan of the database to determine the count of each candidate in C_k would result in the determination of L_k (i.e. all candidate having a count no less than the minimum support count are frequent by definition, and therefore belong to L_k). C_k, however, can be huge, and so this could involve heavy computation. To reduce the size of C_k, the Apriori property is used as follows. Any (K-1)_itemset that is not frequent count is a subset of a frequent k-itemset. Hence, if any (k-1)_subset of a candidate k_itemsets is not in L_{k-1}, then the candidate cannot be frequent either and so can be removed from C_k. This subset testing can be done quickly by maintaining a hash tree of all frequent itemsets. The following steps illustrate the Apriori algorithm [7].

Algorithm: Apriori finds frequent itemset using an iterative level-wise approach based on candidate generation.

Input: Database (D) of transaction;

```
        Minimum support (min-sup)
        Threshold
Output: Association Rules.
Begin
        L1 = find_frequent_1-itemsets(D);
        For (k = 2; Lk-1 ≠Φ; k++)
        Begin
              Ck = apriori_gen(Lk-1, min_sup);
              For each transaction t ∈ D
              Begin
                      Ct = subset(Ck,t);
                      For each candidate c ∈ Ct
                              C.count++;
              End;
                   Lk ={c ∈ Ck |c.count ≥ min _sup}
        End;
        Return L = Uk Lk ;
End.
Procedure Apriori_gen(Lk-1:frequent(k-1)-itemsets;

                           min_sup: minimum support threshold)
Begin
For each itemset l1 ∈Lk-1
```

```
  For each itemset l2 ∈ Lk-1
   If (l1[1] = l2[1]) ∧ (l1[2] = l2[2]) ∧…∧(l1[k-2] =

         l2[k-2])  ∧ (l1[k-1] < l2[k-1] ) Then
   Begin
      c = l1 join  l2;
        If has_infrequent_subset(c,Lk-1) Then
                 Delete c
        Else
                 add c to Ck ;

   End
Return Ck;
End

Procedure has_infrequent_subset (c:candidate k_itemsets);
Begin
     For each (k-1)_subset s of c
           If s ∉Lk-1 then
                         Return TRUE;
Return FALSE;
End.
```

5- The Suggested Approach for Petri Net Reduction Using Association Rules

The suggested approach depends on the logical relations between places and transitions in the Petri net. Association rules techniques are used to extract these logical relations, especially Apriori algorithm. The standard *Apriori* algorithm is not reasonable because it checks k-item set with only (k-1)-item set. We need to check k-item set with all previous item sets. The proposed approach has two main process, these are :

1-Forward process : which includes two stages :
  a. Convert Petri net. to logical program (i.e. production rules).
  b. Apply improved Apriori algorithm to these production rules for reducing the premises.

2- Backward process : which means reconstruct Petri net from the reduced logical program (i.e. reduced production rules).

The following algorithm illustrates the conversion of Petri net to logical program.

```
Algorithm: Conversion of Petri net to logical program.
Input: Petri net
Output: Logical Program
Begin
        i=1
L1 :    For each transition t(i) Do
        If there is no output place from t(i) Then Goto L2;
        Convert the output place p from t(i) to a conclusion of
        clause(i) of the logic program
L2 :    If there is no input place from t(i) Then Goto L4;
        j=1
```

```
L3 :  Add the input place(j) of t(i) as  j^th condition for clause(i)
      j=j+1
If no input place is left Then stop Else Goto L3;

L4 :  i=i+1
If no transition is left in the Petri net Then stop

                Else Goto L1;
End.
```

The algorithm below illustrates the process or logical program reduction using improved Apriori algorithm which is proposed in [1].

Algorithm: **Reduction of logical program using improved Apriori algorithm.**

Input**: Production rules of an expert system.**
Output**: Reduction production rules of an expert system.**

```
Begin

    Convert the premises of production rules into transaction
    database;
    Sort the association rules descending (depending on
    transaction length of right-hand side then left-hand side);

    Repeat
           Replace the symbols of first association rules with one
            symbol in transactions database;
            Delete the rest association rules that contain all symbols
            in the first one;
      Until No association rules;

Recover the production rules from transaction database;
End.
```

The last proposed algorithm is to reconstruct the Petri net from reduced logical program as shown below.

```
Algorithm: Reconstruct Petri net from logical program.
Input: Logical program
Output: Petri net
Begin
    i=1
L1 : For each clause (i) Do
        If there is no conclusion in clause (i) Then Goto L2;
Convert the conclusion of clause (i) to an output place p of t(i)
L2: If there is no condition for clause (i) Then Goto L4:
     j=1
L3:Add the condition(j) of clause(i) as jth input for place(j) of t(i)
   j=j+1
   If no condition is left Then stop Else Goto L3;
L4: i=i+1
   If no clause is left in the logic program Then stop
Else Goto L1;
End.
```

These three proposed algorithms formed the proposed approach to reduce the Petri net using improved Apriori algorithm which converts the Petri net to logical program.

6- Conclusion

The proposed approach provides a precise logical way to convert Petri net to logical program in order to represent an exactly logical method of Petri net. Apriori algorithm provides a good logical reduction for a logical program. The proposed approach depends on conversion of Petri net to a logical program then Apriori algorithm will reduce this program, at last, reconstruct the Petri net from reduced logical program. Petri net reduction is an important process in several Petri net applications, therefore, the proposed approach will play a big role in these applications.

References

[1] Ahmed T. Sadiq, **"*Premises Reduction of Rule-Based Expert Systems Using Association Rules Technique*",**

[2] G. Peterka & T. Murata, **"*Proof Procedure and Answer Extraction in Petri Net Model of Logic Programs*"**, IEEE Transaction software engineering, Vol. 15, No. 2; 1989.

[3] J. L. Petrson**, "*Petri Net Theory and the Modeling of Systems*"** Englewood cliffs, NJ: Prentice-Hall; 1981**.**

[4] Jiawei Han and Micheline Kanmber, **"*Data Mining Concept and Techniques*",** Academic Press, USA, 2001**.**

[5] K. Schmidt, "***Applying Reduction Rules to Algebraic Petri Net***", research reports, Helsinki University of Technology- Finland, No. 4; March 1997.

[6] Matthew B. Dwyer and Lori A. Clarke, "*A Compact Petri Net Representation and its Implications for Analysis***", Software Eng. IEEE Transaction, Vol. 22, Nov. 1996, p. 794-811.**

[7] Rakesh Agrawal and Ramakrishnan Strikant, **"*Fast Algorithms for Mining Association Rules*",** Proc. of the 20th Int'l Conference on Very Large Databases, Santigo, Chile, 1994.

[8] Rakesh Agrawal, **"*Mining Sequential Patterns*",** Proc. of the Int'l Conference Data Mining Engineering (ICDE95), Taiwan, March 1995.

[9] Rakesh Agrawal, Heikki Mannila, Ramakrishnan Strikant, Hannu Toivenen and Inkeri Verkamo, **"*Fast Discovery of Association Rules*",** Proc. of the 20th Int'l Conference on Very Large Databases, Santigo, Chile, 1994.

[10] Rakesh Agrawal, Tomasz Imielinski and Arun Swami**, "*Mining Association Rules Between Sets of Items in Large Database*",** Washington, U.S.A., May 1994.

[11] S. Haddad, **"*A Reduction Theory for Colored Nets*",** Paris University, C.N.R.S. MASI, Paris, France.

[12] T. Murata**, "*Modeling and Analysis of Concurrent System*"**, in Handbook of software Engineering, C. R. Vick and C. V. Rama moorthy, Eds. New York: Van Nostrand Reinhold, 1984, PP.39-63.

[13] T. Murata**, "*Petri Nets: Properties, Analysis and Applications*"** Proceeding of IEEE, Vol. 77, No. 4; April 1989, PP.541-580.

تقليص شبكات Petri باستخدام تقنية القواعد الارتباطية

*أ.م. د. أحمد طارق صادق *اقباس عزالدين علي

المستخلص

هناك عدة طرق لتقليص شبكات Petri التي تلعب دوراً مهماً في تطبيقات واسعة المجال. في هذا البحث سيتم تقديم نموذج جديد ومختلف لتقليص شبكات Petri . النموذج المقترح يعتمد على تحويل شبكات Petri الى برنامج منطقي ومن ثم معالجة هذا البرنامج من خلال تقنية القواعد الارتباطية باستخدام خوارزمية Apriori لاستخراج العلاقات المنطقية لقواعد الانتاج والتي تؤدي الى تقليص البرنامج المنطقي. وفي النهاية يتم اعادة بناء شبكات Petri من خلال البرنامج المنطقي المقلص. اثبتت التجارب التي اجريت على ان الطريقة المقترحة جيدة وكفوءة ومنطقية في عملها لتقليص شبكات Petri.

* قسم علوم الحاسبات/الجامعة التكنولوجية