Research on transition priorities in group based on CPN

Hong Wang[1, 2, 3*], Tao Zhang[2]

[1]*Academy of OPTO-Electronics, Chinese Academy of Sciences, Deng Zhuang Rd. 9, Distinct Haidian, Beijing, China*

[2]*Technology and Engineering Center for Space Utilization, Chinese Academy of Sciences, Deng Zhuang Rd. 9, Distinct Haidian, Beijing, China*

[3]*University of Chinese Academy of Sciences, Yuquan Rd. 19, Shijingshan District, Beijing, China*

Received 1 March 2014, www cmnt.lv

Abstract

Transition priorities might be a useful mechanism when modelling using Petri nets. Accordingly, the newest CPN Tools, widely used for modelling and simulating the Coloured Petri net, implements transition priorities. Whereas, the algorithms compute enabling for all transitions in a highest-priority-first order. In the real system, it is usually that there are priorities relationships not for all transitions but only some ones. Based on the above analysis, this paper put all the transitions, having priority relations, into one group and advances relative theoretical definitions of transition priorities in group, such as absolute preset of transition, key place set, key colour set, etc. Furthermore, it proposes new algorithms when the systems have different key place set and key colour set, and construct the model of the interrupt priorities to solve the problem of software model checking for interrupt system.

Keywords: Petri net, CPN, transitions, transition priorities in group, counter place

1 Introduction

In the process of development of Petri network, in order to better describe realistic system, researchers advanced various high-level Petri net models, such as: time Petri nets [1,2], Stochastic Petri nets [3,4], Colored Petri nets [5,6], Priorities Petri nets [7], Hybrid Petri nets [8], and so on. These extended concepts enhanced Petri net description of capabilities, furthermore, modelling various Petri net had been widely applied in the synchronous system, asynchronous system, simulation [9, 10] and analysis [11, 12] process. However, the actual applications need a better tool to simulate and verity generated models. CPN Tools widen the usage of Petri nets in practical work [13-15].

CPN Tools version 4.0 [16], supporting transition priorities, could specify priority P_HIGH, P_NORMAL or P_LOW [17] for transitions, and could also directly give an integer. Nevertheless, this kind of priorities is static and absolute priority for all transitions, precedence relations, in the actual system, exist among not all transitions but some of them.

Reference [18] proposed to increase one Anti-Place to address the transition priorities, shown in FIGURE 1. The Anti-Place is effective, only if the place had tokens in initial state, however, most systems could not meet the mentioned requirements. Reference [19] used the concept of generalized complement position (Place), changing Petri net system with dynamic priority into the Petri net system with non-priority, but the algorithm was too complex even for constructing a simple system.

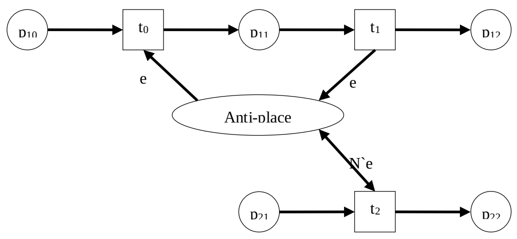

FIGURE 1 Anti-place for transitions with priorities

The remainder of this paper is structured as follows: in the next section, it presented background material and in Section III it provides the definitions of transition priorities in group, key place set and key colour set, etc., and In Section IV, it presents algorithms for increasing one “counter place” to calculate the change of tokens in key place set. Furthermore, the next section, the article solve the practical work interrupt relational model by using the above definitions and algorithms. In Section VI, it concludes and provides directions for future works.

2 Petry net and coloured Petri net

2.1. PETRI NET (PLACE/TRANSITION NET OR P/T NET).

A Petri net is one of several mathematical modelling languages for the description of concurrent systems. A Petri net is a directed bipartite graph, in which the nodes represent transitions and places. The directed arcs described which places are pre- and/or post- conditions, for which transitions (signified by arrows).

***Corresponding author** e-mail: neiep_wh@163.com

A Petri net is a 5-triple $PN=(P,T,F,W,M_0)$ [20], where:

(i) $P=\{p_1,p_2,p_3,...,p_n\}$ is a finite set of places.

(ii) $T=\{t_1,t_2,t_3,...,t_n\}$ is a finite set of transitions.

(iii) $F\subseteq(P\times T)\cup(T\times P)$ is a set of arcs (flow relation).

(iv) $W:F\rightarrow\{1,2,3,...\}$ is a weight initial marking.

(v) $M_0:P\rightarrow\{1,2,3,...\}$ is the initial marking.

$P\cap T=\Phi$ and $P\cup T\neq\Phi$.

The **preset** of a transition or place *x* is the set of its input places or transition: ${}^{\bullet}x=\{y\,|\,(y,x)\in A\}$; its **postset** is the set of its output places or transition: $x^{\bullet}=\{y\,|\,(x,y)\in A\}$.

The **preset** of a set $X_1(X1\subseteq P\vee X_1\subseteq T)$ is the set of its input sets, ${}^{\bullet}X_1=\bigcup_{x\in X_1}{}^{\bullet}x$; its **postset** is the set of its output sets $X_1^{\bullet}=\bigcup_{x\in X_1}x^{\bullet}$.

However, basic Petri nets have limitations. One of the limitations is 'homogenous' tokens: the tokens inside place represent resources; then in Petri nets, all these resources are of the same type and cannot be differentiated. Another limitation of Petri net is that it is not possible to impose additional logical functions ('firing conditions') for a transition.

2.2 COLOURED PETRI NETS (CP-NETS OR CPN)

Coloured Petri nets is one of popular extensions of basic Petri net. In a standard Petri net, tokens are indistinguishable. Nevertheless, in a Coloured Petri Net, each token has a value, which could be descripted with colour set.

A CP-net is a 9-tuple $CPN=(\Sigma,P,T,A,N,C,G,E,I)$ [21], where:

(i) Σ is a finite set of non-empty types, also called colour sets;

(ii) P is a finite set of places;

(iii) T is a finite set of transitions;

(iv) A is a finite set of arcs such that:

$$P\cap T=P\cap N=T\cap A=\Phi;$$

(v) N is a node function. It is defined from A into $P\times T\cup T\times P$;

(vi) C is a colour function. It is defined from P into E;

(vii) G is a guard function. It is defined from T into expressions such that:

$$\forall t\in T:[Type(G(t))=Boolean\wedge Type(Var(G(t)))\subseteq\Sigma].$$

(viii) E is an arc expression function. It is defined from A into expressions such that:

$\forall a\in A:[Type(E(a))=C(p)_{MS}\wedge Type(Var(E(a)))\subseteq\Sigma]$, where *p* is the place of *N*(*a*);

(ix) I is an initialisation function. It is defined from P into closed expressions.

Such that:

$$\forall p\in P:[Type(I(p))=C(p)_{MS}].$$

In popular tools for Coloured Petri nets, such as CPN Tools, the values of tokens are typed, and could be tested (using guard functions) and manipulated with a functional programming language.

3 Definitions of transitions priorities in group

In order to solve transitions priorities in group, the relative definitions listed as follows.

3.1. $P_S(t)$ (TRANSITION PRIORITIES IN GROUP)

When one transition *t* is enabled, there is another transition t_k, $(t_k\neq t)$ could also be enabled, but they have different priorities. A group set is composed of all transitions, which could be enabled at the same time and have different priorities. The group set $S=\{t,t_k\}$, the transition propriety of *t* in group set *S* is called $P_s(t)$. Similarly, the transition propriety of t_k in group set *S* is $P_s(t_k)$.

3.2 T_p (GROUP OF TRANSITIONS WITH PRIORITIES)

When multiple transitions could be enabled at the same time, there is relationship of priorities among transitions in the group. The group is an ordered collection of the sort, according to the priorities from high to low. Defined as follows:

A CP-net with Transitions priorities is a 10-tuple $CPN^*=(\Sigma,P,T,A,N,C,G,E,I,S)$, where:

S is a finite set of group of enabled transitions with priorities. For each, $T_p\in S$,

(i) $T_p\subseteq T, and\,|T_p|>1$;

(ii) $\forall t_i,t_j\in T_p,(j>i)$, transition t_i and t_j could be enabled at the same time, and have different priorities. The priorities of t_i in group T_p is $P_{T_p}(t_i)$, and t_j is

$$P_{T_p}(t_j),\;P_{T_p}(t_i)\geq P_{T_p}(t_j);$$

(iii) In group of transitions with priorities $T_p=\{t_1,t_2,t_3,...\}$, at least one t_i , making $P_{T_p}(t_i)>P_{T_p}(t_{i+1})$.

3.3 $[{}^{\bullet}t_i]$ (ABSOLUTE PRESET OF TRANSITION t_i)

Absolute preset of transition t_i is composed of the places, which are elements of preset of t_i, but not be preset of other transitions in the same group T_p.

$$[{}^{\bullet}t_i] = \{p \mid p \in {}^{\bullet}t_i \wedge (\forall t_k \in T_p(k \neq i), p \notin {}^{\bullet}t_k)\}. \qquad (1)$$

3.4 P_k (KEY PLACE SET)

P_k (Key Place Set) is a set of places, effecting transition priorities in same group. Search algorithm of P_k as follows:

STEP 1: Identify Group of Transitions with Priorities T_p.

STEP 2: Find Key Place Set P_k according to the following Equation:

$$P_k = \{\bigcup_{i=1}^{n} {}^{\bullet}t_i - \bigcap_{i=1}^{n} {}^{\bullet}t_i \mid t_i \in T_p\} - [{}^{\bullet}t_n]. \qquad (2)$$

Firstly, identify preset for each transition t_i, $(t_i \in T_p)$ in group T_p, then, removed public preset of them. Finally, expurgated the absolute preset of transition t_n (with the lowest priority in group T_p).

3.5 $C_k(p)$ (KEY COLOUR SET OF KEY PLACE P) AND $C_k(P_k)$ (KEY COLOUR SET OF KEY PLACE SET P_k)

$C_k(p)$ (Key Colour Set of Key Place p, for short Key Colour Set) is a multi-set of colour set of key place $p(p \in P_k)$. They effected the change of tokens in the key klace p.

Usually, if the key place p has simple colours set $C(p)$, the key colour set of p is same with $C(p)$. Otherwise, p has compound colour set, the key colour set is a multi-set, composed with elements in compound colour set.

$C_k(P_k)$ (Key Colour Set of Key Place Set P_k) is a multi-set of key colour set for each key place p in group P_k.

$$C_k(P_k) = \sum_{k=1}^{n} C_k(p). \qquad (3)$$

Supposing that:

(i) The key place set is $P_k = \{p_1, p_2\}$.

(ii) The colour set of p_1, a simple colour set, is $C(p_1) = c_1$, and the key colour set of p_1 is $C_k(p_1) = c_1$. Length of $C_k(p_1)$ is $|C_k(p_1)| = 1$.

(iii) The colour set of p_2, a compound colour set $C(p_2) = c_1 \cdot c_2 \cdot c_3$, and the key colour set of p_2 is $C_k(p_2) = \{c_1, c_2\}$. Length of $C_k(p_2)$ is $|C_k(p_2)| = 2$.

Then, $C_k(P_k) = \{c_1, c_1, c_2\}$ is a multi-set, and the length of $C_k(P_k)$ is $|C_k(P_k)| = 3$.

3.6 ${}^{\bullet}\overline{p}$ (RELATIVE PRESET OF P) AND $\overline{p}^{\bullet}$ (RELATIVE POSTSET OF P)

p is a key place. The relative preset and postset of p must be set of transitions.

$$\begin{aligned} &{}^{\bullet}\overline{p} = \{t \mid t \in {}^{\bullet}p \wedge (Type(E(a_{in})) \cap C_k(p) \neq \Phi)\}, \\ &a_{in} = (t, p) \in A \end{aligned} \qquad (4)$$

$$\begin{aligned} &\overline{p}^{\bullet} = \{t \mid t \in p^{\bullet} \wedge (Type(E(a_{out})) \cap C_k(p) \neq \Phi)\}. \\ &a_{out} = (p, t) \in A \end{aligned} \qquad (5)$$

4 Algorithms of transitions priorities in group

4.1 ALGORITHM 1: MODEL FOR $|P_k| = 1$ AND $|C_k(P_k)| = 1$

Example 1: FIGURE 2 is a typical of concurrent systems with priority relationships. When t_1 and t_2 are enabled at the same time, t_1 has a higher priority. However, when t_{10} and t_1 are enabled simultaneously, they have no priorities, randomly trigger.

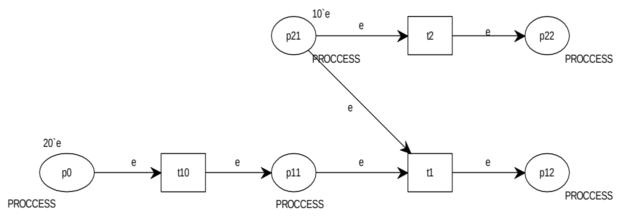

FIGURE 2 Source Model,with $|P_k| = 1$ and $|C_k(P_k)| = 1$

STEP 1: Identify group of transitions with priorities T_p.

In Figure 2, transition set is $T = \{t_1, t_2, t_{10}\}$. t_1 and t_2 have priorities between them, but have no priorities with other transitions. Furthermore, the priority of t_1 is higher than t_2. The group of enabled transitions with priorities is T_p, $T_p = \{t_1, t_2\}$ and $P_{T_p}(t_1) > P_{T_p}(t_2)$.

STEP 2: Identify P_k, the key place set of T_p, and $C_k(P_k)$, the key colour set of P_k.

$^{\bullet}t_1 = \{p_{11}, p_{21}\}$,

$^{\bullet}t_2 = \{p_{21}\}$,

$$So, P_k = \{\bigcup_{i=1}^{2} {}^{\bullet}t_i - \bigcap_{i=1}^{2} {}^{\bullet}t_i \mid t_i \in T_p\} - [{}^{\bullet}t_2] = \{p_{11}\},$$

$C_k(p_{11}) = \{PROCCESS\}$,

$C_k(P_k) = \{PROCCESS\}$.

STEP 3: Modify the CPN model, increasing one "counter place" to calculate tokens in key place set.

STEP 3.1: Increase variable for "counter place".

var n:INT;

STEP 3.2: Increase "counter place" for key place set $P_k = \{p_{11}\}$ in CP-net model, with definition of colour set and initial value.

Increase "counter place" *count*, with colorset *INT*. Because the initial count of the key colour PROCCESS is 0, the initial value of "counter place" *count* is 0.

STEP 3.3: If the relative preset of the key place p is not empty, added two arcs from "counter place" *count* to $^{\bullet}\overline{p}$; and if the relative postset of the key place p is not empty, add two arcs from "counter place" *count* to $^{\bullet}\overline{p}$. Added arcs describe the change of tokens in key place.

Transition t_{10} triggered, place p_{11}. Increase one *PROCCESS*, while t_1 triggered, place p_{11} decreased one *PROCCESS*. So, added two arcs from *count* to its relative preset $^{\bullet}\overline{p_{11}} = \{t_{10}\}$, one is outcoming arc with function "n", the other is incoming arc with function "n+1". Similarly, added two arcs from *count* to its relative postset $\overline{p_{11}}^{\bullet} = \{t_1\}$, one is outcoming arc with function "n", the other is incoming arc with function "n-1".

STEP 3.4: added one bi-directional arc with function "0", from "counter place" *count* to other transitions with lower priorities in the same group.

For $T_p = \{t_1, t_2\}$ in the model, extracted one bi-directional arc from *count* to t_2.

The finished model is shown in Figure 3.

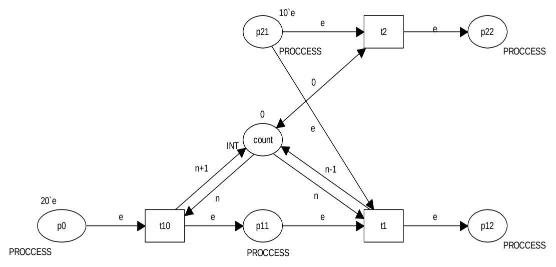

FIGURE 3 Model with priorities $|P_k| = 1$ and $|C_k(P_k)| = 1$

4.2 ALGORITHM 2: MODEL FOR $|P_k| \geq 1$ AND $|C_k(P_k)| > 1$

Example 2: Added two nodes, p_{21} and t_{20}, in FIGURE 2. Transition t_1 could enable, only when p_{11}, p_{21} and p_{31} have tokens in place. Furthermore, t_1 has a higher priority than t_3. However, t_1 and t_3 couldn't plunder privilege of other transitions, t_1 should have same priority with t_{10} and t_{20}, as shown in FIGURE 4.

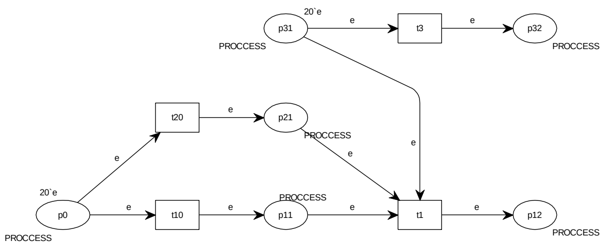

FIGURE 4 Source Model with $|P_k| \geq 1$ and $|C_k(P_k)| > 1$

STEP 1: Identify group of transitions with priorities T_p.

In Example 2, transition set is $T = \{t_1, t_3, t_{10}, t_{20}\}$. The priority of t_1 is higher than t_3. The group of transitions with priorities is T_p, $T_p = \{t_1, t_3\}$, and $P_{T_p}(t_1) > P_{T_p}(t_3)$.

STEP 2: Identify P_k, the key place set of T_p, and $C_k(P_k)$, the key colour set of P_k.

$^{\bullet}t_1 = \{p_{11}, p_{21}, p_{31}\}$,

$^{\bullet}t_3 = \{p_{31}\}$,

$$P_k = {}^{\bullet}t_1 \cup {}^{\bullet}t_3 - {}^{\bullet}t_1 \cap {}^{\bullet}t_3 - [{}^{\bullet}t_3] = \{p_{11}, p_{21}, p_{32}\} - \{p_{31}\} - \Phi = \{p_{11}, p_{21}\}.$$

Furthermore:

$C_k(p_{11}) = \{PROCCESS\}$,

$C_k(p_{21}) = \{PROCCESS\}$.

So: $C_k(P_k) = \{PROCCESS, PROCCESS\}$.

STEP 3: Modify the CPN model, Increase a "counter place" to calculate tokens in key place set.

STEP 3.1: Increase variables for "counter place" and trigger function for transition with lower priorities.

According to $C_k(P_k) = \{PROCCESS, PROCCESS\}$ and $|C_k(P_k)| = 2$, added a "counter place" with 2-tuples in the model for p_{11} and p_{21}. Defined variables and trigger function by CPN ML as follows:

```
//new colorset for "counter place" with two-tuples
        colset INTLIST = product INT * INT;
        var n1,n2:INT;
        fun tran(counts: INTLIST) =
let
    val count1 = #1 (counts);
    val count2 = #2 (counts);
in
//modify expression according actural condition
    not(count1 <>0 andalso count2 <>0)
end
```

STEP 3.2: Increase "counter place" for key place $P_k = \{p_{11}, p_{21}\}$ in CP-net model, with definition of colour set and initial value.

Increase "counter place" *count*, with colour set INTLIST. Because the initial count of the key colour PROCCESS in p_{11} and p_{21} are all 0, the initial value of "counter place" *count* is (0,0).

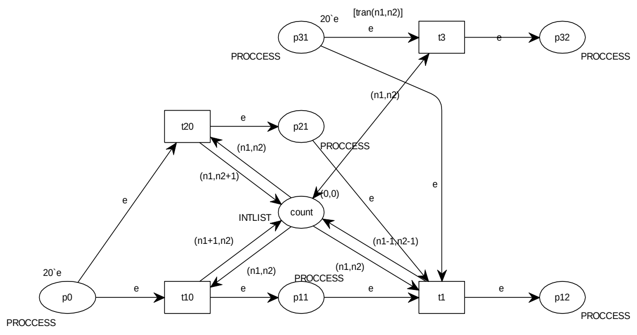

FIGURE 5 Model with priorities with $|P_k| \geq 1$ and $|C_k(P_k)| > 1$

STEP 3.3: Do as follows for each $\forall p \in P_k$: if the relative preset of the key place p is not empty, add two arcs from "counter place" *count* to $^{\bullet}\overline{p}$; and if the relative postset of the key place p is not empty, add two arcs from "counter place" *count* to $\overline{p}^{\bullet}$. Added arcs describe the change of tokens in key place *Count*, the "counter place", looks like the format (n_1, n_2) . And n_1 showed the change of p_{11} , n_2 records the change of p_{21} . Furthermore, changed in adding arcs and the arc functions (as shown in Figure 5).

STEP 3.4: If there is no outcoming arc from "counter place" *count* to other transitions with lower priorities in same group, added one bi-directional arc with function (n_1, n_2) .

STEP 3.5: Added trigger function in all transitions with lower priorities.

In this example, added trigger function $tran(n_1, n_2)$ for transition t_3 .

5 Practise example

Example 3: The systems have a main program Proccess, and two external interrupts *INTR₁* and *INTR₂*, which have the priorities relationships that the priorities of *INTR₁* is higher than *INTR₂*, namely $P_{INTR1} > P_{INTR2}$. The question is how to construct a CPN model to meet interrupts priorities.

The main process *Proccess* set interrupts on or off. The place *INTR* managed all the interrupts that is, when *Proccess* set the interrupt on, it send semaphore *intr₁* to transition *on_intr₁* in order to enable the active interrupt. Otherwise, when it set the interrupt off, it remove *intr₁* from transition *on_intr₁*, thus disabled the interrupt, as shown in Figure 6.

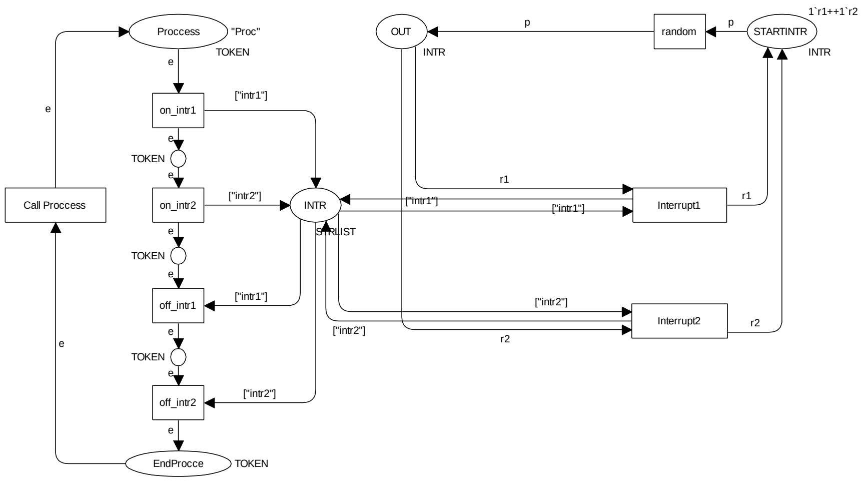

FIGURE 6 Interrupt model M_3 without priorites

The place *STARTINTR* simulated the external interrupts, the transition *random* control the happening of interrupts randomly, and the place *Out* represented the current enabled interrupts. The interrupt conditions are:

(i) *INTR₁* had been enabled;

(ii) The existence of the external interrupt *INTR₁*, then *Interrupt₁* could run. However, the model could be shown as FIGURE 6, although given the current disconnect, turn control to an external interrupt occurs, but not showing the priorities of relations between *INTR₁* and *INTR₂*.

If omitted other unrelated contents, the interrupt model could be minimized, as shown in Figure 7.

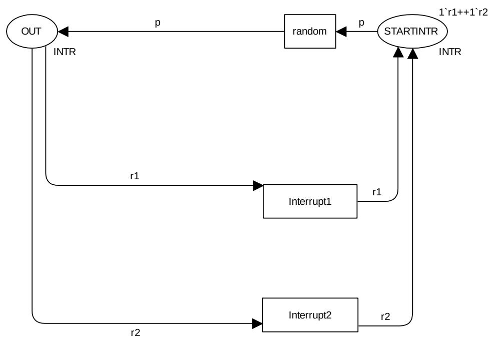

FIGURE 7 Minimized interrupt model M$_4$

The minimized interrupt Model *M$_4$* use the above algorithm, as show as Section IV.

STEP 1: Identify group of transitions with priorities T_p.

In Figure 7, transition set is $T = \{random, Interrupt_1, Interrupt_3\}$. The group of enabled transitions with priorities is T_p.

$T_p = \{Interrupt_1, Interrupt_2\}$,

So, $P_{T_p}(Interrupt_1) > P_{T_p}(Interrupt_2)$.

STEP 2: Identify P_k, the key place set of T_p, and $C_k(P_k)$, the key colour set of P_k.

$^{\bullet}Interrupt_1 = \{out\}$,

$^{\bullet}Interrupt_2 = \{out\}$.

Thus,

$$P_k = {^{\bullet}Interrupt_1} \cup {^{\bullet}Interrupt_2} - {^{\bullet}Interrupt_1} \cap {^{\bullet}Interrput_2} - [{^{\bullet}Interrupt_2}] = \Phi .$$

Namely, it became an unsolvable problem by using the above algorithm.

Secondary transitions, *call_intr$_1$*, *call_intr$_1$*, and places, *OUTING$_1$*, *OUTING$_2$*, could be introduced to the previous model *M$_4$*. Furthermore, place *OUT* would be divided into two places *OUT$_1$* and *OUT$_2$*. Thus, the model *M$_4$* could be changed as *M$_4$'*, shown in Figure 8.

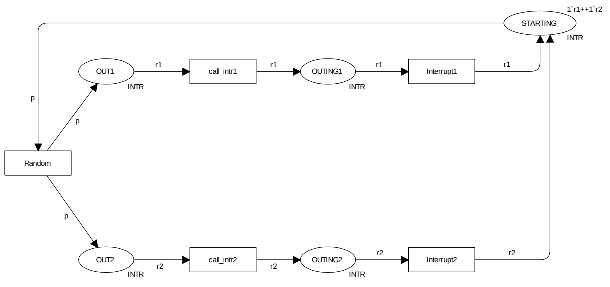

FIGURE 8 Auxiliary Model M$_4$'

$^{\bullet}Interrupt_1 = \{OUTING_1\}$,

$^{\bullet}Interrupt_2 = \{OUTING_2\}$.

Thus,

$$P_k = {^{\bullet}Interrupt_1} \cup {^{\bullet}Interrupt_2} - {^{\bullet}Interrupt_1} \cap {^{\bullet}Interrput_2} - [{^{\bullet}Interrupt_2}] = \{OUTING_1, OUTING_2\} .$$

Next,

$C_k(OUTING_1) = \{INTR\}, C_k(OUTING_2) = \{INTR\}$

So, $C_k(P_k) = \{INTR, INTR\}$.

In the model *M$_4$'*, when the places, *OUTING$_1$*, and *OUTING$_2$*, being changed, they could affect trigger priorities of transitions *Interrupt$_1$* and *Interrupt$_2$.*

STEP 3: Modify the CPN model, increasing one "counter place" to calculate tokens in key place set, as shown in Figure 9.

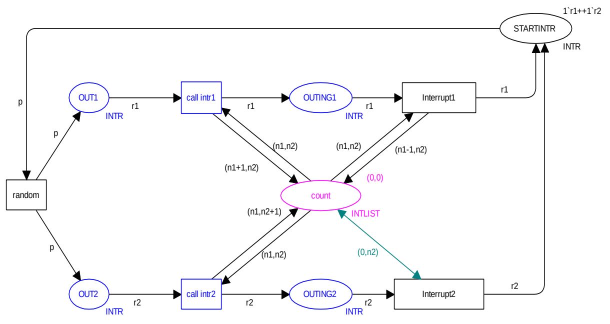

FIGURE 9 Partial interrupt model with priorities

Add the modified partial model to the original model to form a complete interrupt model with priorities *M$_5$*, as shown in Figure 10.

In the complete model, the main process *Proccess* could send semaphore to the place *INTR* to control whether the interrupts are enabled. Moreover, the transition *Interrupt$_1$* or *Interrupt$_2$* could be tangled only when outing interrupts signal is enabled. Meantime, only the transition *Interrupt$_1$* could be triggered although it and *Interrupt$_2$* could be enabled simultaneously, namely, *Interrupt$_1$* have higher priorities than *Interrupt$_2$*.

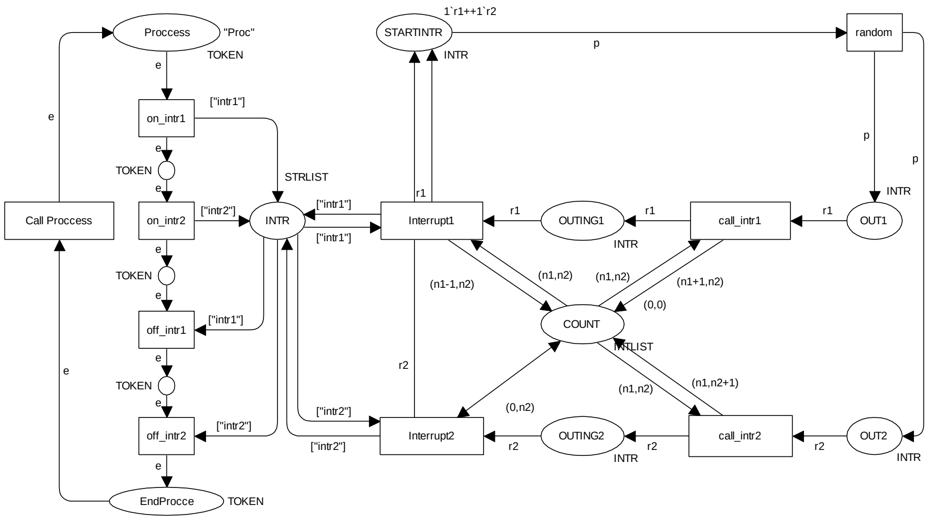

FIGURE 10 Complete interrupt model with priorities

6 Working outlook

This paper researched the CPN model with transition priorities in Group. When the original model has lower complexity, that was, $|P_k|$ and $|C_k(P_k)|$ is small, algorithm for transition with priorities is better. However, this algorithm only addressed transition priorities under enabling condition, not under certain conditions. After that, it could expand the research work in this regard. Furthermore, due to "count place" need to connect bi-directional arc to multiple transitions, making the more complex model structure, reducing the complexity of the model is also the focus of future research.

References

[1] Bożek A 2012 Using Timed Coloured Petri Nets for Modelling, Simulation and Scheduling of Production Systems *Production Scheduling ed Righi R D R: Production Scheduling* 24
[2] Louchka P-Z 2013 Timed Petri Nets. *Time and Petri Nets Springer Berlin Heidelberg* 72
[3] Rogge-Solti A, Weske M 2013 Prediction of remaining service execution time using stochastic petri nets with arbitrary firing delays *Service-Oriented Computing Springer Berlin Heidelberg* 15
[4] Lohmann N, Song M, Wohed P 2014 Discovering Stochastic Petri Nets with Arbitrary Delay Distributions from Event Logs *BPM 2013 ed Lohmann N et al Springer International Publishing Switzerland* 13
[5] Jensen K, Kristensen L M, Lisawells 2007 Coloured Petri Nets and CPN Tools for modelling and validation of concurrent systems. *International Journal on Software Tools for Technology Transfer* **9**(3) 213-54
[6] Gehlot V, Nigro C 2010 An introduction to systems modelling and simulation with colored Petri nets *2010 IEEE Proceedings of Simulation Conference* 104-18
[7] Boukhentiche H, Abbas-Turki A, Moudni A E 2011 On the bus priority dilemma: A Petri Net model with resource sharing and inhibitor arc *2011 IEEE International Conference on Industrial Technology* 88-93
[8] Balduzzi F, Giua A, Menga G 2000 First-Order Hybrid Petri Nets: A Model for Optimization and Control *IEEE Transactions on Robotics and Automation* **16**(4) 382-99
[9] Javan A, Akhavan M, Moeini A 2013 Simulating Turing Machines Using Colored Petri Nets with Priority *Transitions. International Journal on Recent Trends in Engineering & Technology* **9**(1) 75-81
[10] Van der Aalst W M P, Stahl C, Westergaard M 2013 Strategies for Modelling Complex Processes Using Colored Petri Nets *Transactions on Petri Nets and Other Models of Concurrency VII, ed Jensen K et al: Springer Berlin Heidelberg* 50
[11] Karamti W, Mahfoudhi A, Kacem Y H 2012 Hierarchical Modelling with Dynamic Priority Time Petri Nets for Multiprocessor Scheduling Analysis *2012 international conference on embedded systems and applications* 114-21
[12] Jensen K, Kristensen L M, Lisawells 2007 Coloured Petri Nets and CPN Tools for modelling and validation of concurrent systems *International Journal on Software Tools for Technology Transfer* 9 213-54
[13] Asmuss J, Lauks G and Zagorskis V 2012 Application of CPN Tools for Simulation and Analysis of Brandwidth Allocation *World Academy of Science, Engineering and Technology* 6 72-5
[14] Westergaard M, Fahland D, Stahl C 2013 Grade/CPN a Tool and Temporal Logic for Testing Colored Petri Net Models in Teaching *Transactions on Petri Nets and Other Models of Concurrency VIII, ed M. Koutny et al: Springer Berlin Heidelberg* 23
[15] Westergaard M, Kristensen L M 2009 the Access CPN Framework-A Tool for Interacting with the CPN Tools Simulator *Applications and Theory of Petri Nets Springer Berlin Heidelberg* 10
[16] Home Page of the CPN Tools, http://cpntools.org. Accessed on September 22th, 2013.
[17] Westergad M, Verbeek H M W 2011 Efficient Implementation of Prioritized Transitions for High-level Petri Nets *PNSE'11 - Petri Nets and Software Engineering* 27-41
[18] Sheng-De W, Wang-Bin H, Zhong-Chang X 2008 Research On Simulation Model of Petri Net with Priority by CPN Tools *Journal of System Simulation* **20**(3) 814-6
[19] Wen-Jun L, Xiao-Chong Z, Shi-Xian L, Jian M 2001 Petri Net Semantics of Dynamic Priority Systems *Chinese J. Computers* **24**(10) 1085-94
[20] Murata T 1989 Petri Nets: Properties, Analysis and Applications *Proceedings of the IEEE* **77**(4) 541-80
[21] Jensen K 1994 An Introduction to the Theoretical Aspects of Coloured Petri Nets *A Decade of Concurrency Reflections and Perspectives Springer Berlin Heidelberg* 230-72

Authors

Hong Wang, born in March, 1978, Tieling, Liaoning Province, China

Current position, grades: PhD candidate at Technology and Engineering Center for Space Utilization, Chinese Academy of Sciences, a lecturer in the School of Control and Computer Engineering, North China Electric Power University.
University studies: computer science technology and application.
Scientific interest: high reliability software testing and model checking.
Publications: 3 papers.

Tao Zhang, born in December, 1972, Zhaozhuang, Shandong Province, China

Current position, grades: professor at Technology and Engineering Center for Space Utilization, Chinese Academy of Sciences.
University studies: PhD at Tsinghua University in China.
Scientific interest: high reliability software testing and validation, high reliability electronic information system analysis, design methods, complex system simulation.
Publications: more than 20 papers.