行政院國家科學委員會專題研究計畫　成果報告

MRDFG 的周期界的計算的提升

計畫類別： 個別型計畫
計畫編號： NSC94-2213-E-004-005-
執行期間： 94 年 08 月 01 日至 95 年 07 月 31 日
執行單位： 國立政治大學資訊管理學系

計畫主持人： 趙玉

報告類型： 精簡報告

處理方式： 本計畫可公開查詢

中　華　民　國 95 年 10 月 13 日

MRDFG 的周期界的計算的提升

計畫編號：NSC 94-2213-E-004-005
執行期限： 94 年 8 月 1 日至 94 年 7 月 31 日

主持人：趙玉 政治大學資管系
計畫參與人員：

一、中文摘要

Schoenen [4]證實了我們的理論。即如果 MRDFG 的標記使它像 SRDFG 一樣表現,不需要變換為 SRDFG 。 他們表明當標記高于相符標記,在回界相對於標記的圖中,相同圖案重複周期性出現。 我們提議延長我們理論對此提供一解釋並且發展理論將展開具有環結合 MRDFG 展開成進等效的 MRDFG 而無須 轉換為 SRDFG 以及進一步轉變成具有相同弧線和節點等效的 SRDFG。 因此比現有的技術好。 當 MRDFG 不具有活性時,變換或者展開浪費時間和記憶住。 我們提議用[11]裡理論來研究這樣的一個問題。

關鍵詞：彈性生產系統, 派翠網, 死結, 虹吸

Abstract

Schoenen [4] confirmed our theory that if the marking of an MRDFG makes it behave like an SRDFG, there is no need for the conversion to SRDFG. They indicated that above CM (Comfortable Marking), the same pattern repeats plus one full throughput step of value one and so on. We propose to extend on our theory to offer an explanation and develop theory to unfold MRDFG with loop-combination into equivalent MRDFG without SRDFG conversion and further to convert it into equivalent SRDFG with the same set of arcs and nodes. Hence it is better than existing techniques. Conversion or unfolding wastes time and memory when the MRDFG is not live. We propose to study such an issue based on the theory in [11].

Keywords: **Flexible manufacturing systems, Deadlock prevention, Petri nets, Siphons, S^3PR.**

二、緣由與目的

1. Several multi-rate signal processing algorithms can be conveniently represented by multi-rate data-flow graphs [12] which are equivalent to General Petri nets (GPN). GPN is useful for modeling flexible manufacturing systems with multiple robots and workstations [13] and for parallel programs [10]. MRDFG (Multi-Rate Data-Flow Graph) differs from SRDFG (Single-Rate Data-Flow Graph) in that the execution of a node may require more than one data from each input arc and may produce more than one output data to each output arc. Thus, unlike SRDFG which can be transformed into Marked Graph, MRDFG can be transformed into weighted Marked Graph where arcs may carry multiple weights. The model of MRDFG is useful for describing some algorithms of communications and signal processing involving interpolation and decimation operations [1,15]. WMG is a natural generalization of marked graph and also a special class of General Petri nets (GPN) [9] where each place holds single input and single output transitions. The WMG is useful for modeling bulk arrivals and services [10] and automatic manufacturing systems [7].

Properties of GPN can be found in Lien [14], Teruel et al [5] and Landwebes et al [16]. We merely cite their results. Lien studied the termination properties (whether any transition can fire infinitely often) of special classes of GPN; i.e., whether they are consistent/conservative. Landwebes et al [16] characterized reachability sets of conflict-free and persistent PN. He developed an exponential time algorithm for deciding boundedness of such nets and determined that it takes exponential space. Teruel et al [5] provided much more comprehensive studies (reachability, liveness, boundedness, consistent, conservative, and others) of GPN. To provide algorithms with less time complexity, attention is limited to subclasses of GPN.

Digital signal processing algorithms are repetitive in nature. These algorithms are described by iterative data-flow graphs where nodes represent computations and edges represent communications. For all data-flow graphs, there exists a fundamental lower bound on the iteration period referred to as the iteration bound. Determining the iteration bound for signal processing algorithms described by iterative data-flow graphs is an important problem.

There are two existing algorithms for determination of the iteration bound [2,6,17]. [12] proposed another novel method based on the minimum cycle mean algorithm to determine the iteration bound with a lower polynomial time complexity than the two existing techniques. It is convenient to represent many multi-rate signal processing algorithms by multi-rate data-flow graphs. The iteration bound of a multi-rate data-flow graph (MRDFG) can be determined by considering the single-rate data-flow graph (SRDFG) equivalent of the MRDFG.

However, first, the equivalent SRDFG contains many redundancies with respect to nodes and edges. Second, it

increases the memory requirement by duplicating nodes and arcs which further prolongs the time for the IB (iteration bound) calculation of the equivalent SRDFG. The iteration bound of the MRDFG can be determined faster if these redundancies in the equivalent SRDFG are first removed. [12] combined elimination of node and edge redundancies to propose a novel algorithm for faster determination of the iteration bound of the MRDFG. This, however, incurs extra overhead and it only improves the efficiency of IB computation. Before the reduction, the increased memory requirement is still required to generate the equivalent SRDFG.

It is generally an exponential time task to transform MRDFG into SRDFG [3] and the transformed SRDFG is much larger (grows exponentially) than the MRDFG. Further, efforts to find the bound are wasted if the MRDFG is not live. Also their algorithm considers only the case of resource-constrained scheduling. In order to execute operations of the processing algorithm in parallel, the required number of processors or functional units required to execute the operations in parallel may be larger than the number of available resources. In that case, they have to give the order of executions, or the precedence, to these operations to reduce the parallelism.

However, modern VLSI is able to put several hundred processors in a single chip like *thinking machines* and resource is no longer a limitation. In this case, the algorithm proposed previously no longer holds. We propose to develop an algorithm for such resource-rich situations and being able to handle resource-constrained cases under small twists.

In [7,8], we extended the technique to determine the IB for SRDFG to that for an MRDFG meeting some sufficient condition where the MRDFG behaves like an SRDFG. There is no need to take the extra time to remove the node and arc redundancy.

We now propose to extend this technique to cases involving loop combinations with redundant nodes and edges much fewer than that in [3]. Loop combination refers to the fact that the IB of an MRDFG may be a linear additive combination of the loop bounds (LB's) of more than one loop. We propose to convert it to another MRDFG (with some node and arc duplications) satisfying the above sufficient condition with no loop combination.

2. We also propose to answer a question by Schoenen [4] as in Figure 2 of [4] which shows the normalized throughput as a function of the WST (weighted sum of tokens). The linear dependency is the bound confirming our theory in [8]. Above CM (Comfortable Marking), the same pattern repeats plus one full throughput step of value one and so on. We could consider decompose one cycle into several with integer multiples of CM plus one with the remaining delays, the throughput of them all accumulates algebraically. **This explains the motif behind the IB study of MRDFG with loop-combinations.** They observed from numerous simulations that a certain kind of point symmetry exists around a certain WST. They were not able to explain this. It seems like a kind of nonexistent-delay-brake that slows down the system if we have less than CM delays. **We propose to apply our theory in [8] to explain this. Due to the symmetry, it is interesting to study the behavior around the symmetry point which has not been treated in any literature.**

3. Schoenen also stated that the liveness condition has to be checked to obtain the zeroes up to WST = 7, otherwise erraneous (but valid) bounds might be introduced. None has considered this problem. As illustrated in [11], the problem for minimum live weighted marking is equivalent to the problem if there is a positive solution of a Diophantine equation, which takes exponential time. We propose to investigate this issue and develop an efficient algorithm. None has investigated the possibility of determining IB for a weighted loop with self loop-combination in an algebraic way. It is interesting to investigate the IB when it is minimum (and above) marked to be live.

三、Results

I. Computation of IB of a Weighted Circuit (WC)

The problem is in general the NP-hard; however, when it is conservative and consistent, the loop is bounded and it must return to its initial marking in finite number of runs. Thus, we attempt to find the best lower bounds followed by repeating reducing the bound to test the liveness until it is dead. The testing also allows us to find the IB for the minimal marking. We have added one new – algebraic approach — in addition to the centralized and distributed ones — to find the lower bound. We find that the distributed method produces lower bound than the centralized one because the former fires transitions more efficiently. However, it is more difficult to estimate the lower bound in an algebraic way. The new algebraic approach derives lower bounds in terms of formulas as follows.

Thus, we consider only the case where only p_1 holds tokens. Define the corresponding weighted sum of tokens W ($m_0(p_1)$, respectively), W_s, as the minimum W (marking at p_1, respectively) to support one system iteration, i.e., $m_0(p_1)=a_o(p_1)*R_0$.

It, however, may overestimate the minimum W, W_{min}, for liveness. Note that one loop iteration is a minimum legal firing sequence that includes every transition in the loop. It is minimal in the sense that any shorter firing sequence is not legal.

Definition: Let N be a Weighted Circuit (WC), W_l (W_s, respectively) is the minimum W to support one system iteration (a consecutive loop iterations to reach one system iteration, respectively). W_{min}, is the minimum W for liveness.

Lemma 1: For a circuit, $W_s \geq W_l$.

Proof: Obvious from definitions of W_s and W_l and the fact that it takes more tokens to support one system iteration than to support a loop iteration. ■

Notice that we do not put $W_D > W_s$ in Lemma 1.

Lemma 2: Let M_0 be a live initial marking for a WC, and $\forall M \in R(M_0)$, if $m(p_1)>0$, then $M_0' = M_0 - [1\ 0\ \dots\ 0]^T$ is also a live marking.

Proof: For all reachable markings, there is at least a token at p_1. Hence we can remove this token without affecting the firing of all transitions. ■

Corollary 1: For W_l of a WC (weighted circuit), $\exists M, M' \in R(M_0)$, $m(p_1)=0$, $m'(p_1)=a_o(p_1)$ and only t_1 is firable at M'.

Proof: If $\forall M \in R(M_0)$, $m(p_1) \geq c$, a positive integer, then $M_0'' = M_0 - [c\ 0\ \dots\ 0]$ is also a live marking and $\exists M$, $M' \in R(M_0'')$, $m(p_1)=0$ and $M'[t1>M, m'(p1)=a_o(p_1)$. $W = m_0''(p_1) * y_1 = W_l$ is minimum in the sense that it is not live for $M_0''' = M_0'' - [1\ \dots 0]$. ■

Corollary 2: For W_l and W_D of a WC, $W_D + y_1 \geq W_l$.

Proof: It becomes live by adding a token at p_1 when $M_0' = M_D = [a_o(p_1)-1,\ a_o(p_2)-1, \dots,\ a_o(p_K)-1]^T$ with W increased from W_D to $W_D + y_1$ and $m(p_1)=a_o(p_1)$, only t_1 is firable. It is live at $W = W_l$ and $\exists M' \in R(M_0)$, $m'(p_1)=a_o(p_1)$ and only t_1 is firable; the number of places having tokens is no fewer than the case where $W = W_D + y_1$. Hence we have $W_D + y_1 \geq W_l$. ■

To find W_{min}, we should find the minimum among $W_D + y_1$, W_l, and W_s.

Theorem 1: For a WC in a WMG, $W_{min} = W_l$.

Proof: By Lemma 1 (Corollary 1, respectively), we have $W_s \geq W_l$ ($W_D + y_1 \geq W$, respectively). The fact that it is not live if $W < W_l$ implies that $W_{min} = W_l$. ■

Estimation of W_l

The maximum W for a dead WC occurs when $m_0(p_j) = a_o(p_j) - 1$, $\forall j$ and all transitions are dead. It is live by adding a token to any p_j. Such a W_D may overestimate the minimum W required for liveness. Recall that we consider only the case where all tokens sit at p_1. We need to find the maximum of $m(p_j)$ $\forall j$ without firing its output transition t_j.

Let g_i be the greatest common denominator (gcd) of $a_i(p_j)$ and $a_o(p_j)$; i.e., $g_j = gcd\ (a_i(p_j),\ a_o(p_j))$. $u_j = max_{k \in \mathbb{IN}}(k * g_j \% a_o(p_j))$, where "%" is the modulus function, "$\mathbb{IN}$" the set of all integers and $j \neq 1$. u_j is the largest integer smaller than $a_o(p_j)$ and is an integral multiple of g_j. $u_1 = a_o(p_1) - 1$.

Lemma 3: 1) $m(p_j) = k * g_j$, $k \in \mathbb{IN}$; 2) maximum of $m(p_j) = u_j$, $\forall j \neq 1$ where t_j is not enabled.

Proof: 1. After t_{j-1} and t_j fire k_1 and k_2 times respectively, $m(p_j) = k_1 * a_i(p_j) - k_2 * a_o(p_j) + m_0(p_j) = g_j * (k_1 * a_i(p_j)\ g_j - k_2 * a_o(p_j)/g_j) = g_j * k$, where $k = k_1 * a_i(p_j)/g_j - k_2 * a_o(p_j)/g_j$. 2) $m(p_j) = k * g_j \% a_o(p_j)$ $\forall j \neq 1$ where t_j is not enabled. Note $m_0(p_j) = 0$. Hence, maximum of $m(p_j) = max_{k \in \mathbb{IN}}(k * g_j \% a_o(p_j)) = u_j$. ■

Theorem 2: 1) The WC is not live iff $W \leq W'_D$. 2) $W_l = (z/y_1) * y_1$ where $z = y_1 + W'_D$, $W'_D = \Sigma_i (y_i * u_i)$.

Proof: 1.(←) Suppose some transitions are firable. Fire all enabled transitions in the WC until all are dead. Then $W' \leq W'_D$ where W' is the W *at the dead marking*. Since W is conserved, we have that the WC is dead if $W \leq W'_D$. (→) Consider the maximum W such that $W \leq W'_D$ (hence; $W + y_1 > W'_D$). Since W must be an integral multiple of y_1, $W = y_1 * (a_o(p_1) - 1) + W''_D$, where W''_D is an integral multiple of y_1. Hence $W''_D \leq W'_D - y_1 * (a_o(p_1) - 1)$ and $W = W'_D / y_1 * y_1$ since W is the maximum such that $W \leq W'_D$. After we add a token to p_1 (new $W' = y_1 + W > W'_D$), the net becomes live. Hence it is live if $W > W'_D$ and if it is not live, then $W \leq W'_D$. 2) The minimum W for liveness is $W_l = y_1 + W'_D / y_1 * y_1 = (z/y_1) * y_1$. ■

In general, we can divide the WMG into a number of groups such that *each group, denoted G_j, has a place, denoted p_j, that all circuits in the group pass through*. Theorem 2 can be extended:

Theorem 3: For a G_i with all tokens at p_j, $W_l = max_j\ W_{lj}$; i.e., W_l is the maximum of W_{lj} of all circuits.

Proof: First if some transitions are not live, so are all transitions. Otherwise, output places (whose output transitions are dead) of live transitions will become unbounded. This is impossible because the net is conserved. For all dead transitions, there is at least one input place p whose $m(p) < a_o(p)$. We trace from such a p backward to its input transition. Repeat this trace process. Due to the finite size of the net, we will reach a transition traced earlier; thus forming a circuit where $m(p) < a_o(p)$ for all places. W for the circuit is less than its W_l — a contradiction. ■

Notice that W_l varies with the place p_1 selected, so does W_{min}. Thus, in order to find the minimum W for liveness, we have to compute W_{min} for all places and select the minimum.

II. Computation of IB for a WMG

We propose to compute the iteration bound (IB) of a WMG by converting the WMG into an equivalent MG=(P', T', F', M_0, W'). It consists of the following steps: 1) decomposing it into a number of loop-combination free islands (lcf-islands), 2) searching the loops and transitions that are involved in loop-combinations, 3) expanding the corresponding loops and combination transitions, 4) determining the IB of each lcf-islands, and 5) selecting the maximum IB in Step 4 as the IB of the WMG. Each step is explained as follows.

0) Using graph-transversal algorithms to divide the net into separate loop-combination free islands (lcf-islands).

Definition: A loop-combination free island (lcf-island) is a strongly connected subnet with no loop-combinations.

Definition: $L_f(p_r)=\{L|\ p_r \in L\}$ is the set of all the elementary (forward) loops L passing through a common place, called the reference place p_r. N is said to be covered by $L^1_f(p_r^1)$, $L^2_f(p_r^2)$, …, $L^k_f(p_r^k)$, if $\forall i, j$, $L^i_f(p_r^i)\cap L^j_f(p_r^j) \neq \Phi$, *iff* $i \neq j$ and $L_N=\{L|\ L$ is an elementary (forward) loop in $N\}= L^1_f(p_r^1)\cup L^2_f(p_r^2) \cup \ldots \cup L^k_f(p_r^k)$. $L_r(p_r, p'_r)=\{L|\ p'_r \in N, \forall L' \in L_f(p_r), p'_r \notin L', L \in L_f(p'_r), \exists L_1 \in L_f(p_r), L_1 \cap L \neq \Phi\}$ is the set of all elementary (reverse) loops that pass through a reference node p'_r not on any loop in $L_f(p_r)$. L_2 is a reverse loop with respect to L_1 if $\exists p_1, p_2 \in N, p_1 \neq p_2, L_2 \in L_r(p_1, p_2), L_1 \in L_f(p_1), L_1 \cap L_2 \neq \Phi$.

Lemma 4: Any L_f is a loop-combination free island (lcf-island).

Proof: By the definition of loop-combination, $\exists t_1 \in L_1, t_2 \in L_2, L_1 \in L_f, L_2 \in L_f$ and a firing sequence σ where t_2 may firing more than once without firing t_1. This is impossible since both L_1 and L_2 pass through p_r.

It takes $O(n)$ to find one lcf-island where $n=|\ T| +|P\ |+|F|$. In the worst case, there are $O(n)$ lcf-islands. The total time complexity is hence $O(n^2)$. Because each lcf-island contains at least one place not in another lcf-island, the number of lcf-islands is $O(|P\ |)$.

1) For each lcf-island I_i, find its critical loop mark factor *clmf using* procedure *clmf* (I_i).

Definition: I_i is *single-enabled* (*multiple-enabled*) if there is only (more than one) one transition is initially enabled in I_i under M_0.

To simplify the presentation, we assume that I_i is *single-enabled*. However, the results also apply to the case of multiple-enabled I_i using the retiming technique.

Definition: Let $(\upsilon)_i$ denote the υ (=t, p, R, …) for island I_i. Let t be a transition in I_i. Define the *loop factor* l_i of I_i to be $l_i=R(t)/R_i(t)$, indicating the number of island iterations (or loop combinations) needed to complete one system iteration. Define the *total enabling factor* e^t_i of I_i to be such that the initially enabled transition t in I_i is enabled $e^t_i R_i(t)$ times under M_0. e^t_i is the number of island iterations that can be performed by firing one round of all transitions in I_i to return to M_0. We say that I_i is said to be enabled e^t_i times.

In general, for a multiple-enabled I_i, e^t_i is determined by the marking in the critical loop similar to the way iteration bound is determined except that the distance is replaced by setting distance from p_a to p_b, $d_{ab}=M_0(p_b)/(a_o(p_b)R_i(t_k))$ if $t_k \in p_a\bullet \cap \bullet p_b$ else $d_{ab}=0$. Then, find the minimal or shortest distance from a node to itself d_{kk} among all places and we have $e^t_i=\min_k d_{kk}$.

procedure *clmf* (I_i)

{

1. compute $l_i= R(t)/R_i(t)$
2. in the Floyd Matrix, set $d_{ab}=M_0(p_b)/(a_o(p_b)R_i(t_k))$ if $t_k \in p_a\bullet \cap \bullet p_b$ else $d_{ab}=0$, $\forall p_a, p_b \in P_i$
3. find the shortest d_{kk}
4. set $e^t_i=\min_k d_{kk}$
5. compute $clmf_i=l_i-e^t_i+1$

}

The total time complexity of this technique is $O(|P\ |^3)$.

2) Duplicate each node in I_i $clmf=l_i-e^t_i+1$ times using procedure *duplicate*(I_i).

Notice that when $e^t_i<l_i$, the island is unable to complete one system iteration by firing all transitions in one round and return to M_0. $l_i-e^t_i$ indicates the corresponding deviation and in order to complete one system iteration, $l_i-e^t_i$ additional rounds must be performed. Thus, we need to duplicate I_i $l_i-e^t_i$ times; this plus I_i itself constitutes $l_i-e^t_i$ duplicates of I_i. In the sequel, let $\{t_\alpha\}=\bullet p_r$. $w=F(t_\alpha, p_r)$.

Definition: t^k is the duplicate of t for the k-th firing of t. $t^{a,b,\ldots v}$ is the duplicate of t for the a-th, b-th, …, and v-th firing of t indicating that $t^a=t^b=\ldots=t^v$. t is called to be incompatible if $\exists I_i, I_j, I_i \neq I_j$, $(t^{k-1})_i=(t^k)_i$, $(t^{k-1})_j \neq (t^k)_j$. The set of incompatible transitions in N is denoted by T_I.

procedure *duplicate*(I_i)
{
1. if $clmf=l_i-e^t_i+1>0$ then {
2. delete the input arc of p_r; i.e., set $\bullet p_r=t_\alpha\bullet$=null.
3. duplicate the new I_i $q=clmf$ times creating I_i^1, I_i^2, …, I_i^q with P_i^j, T_i^j, F_i^j in each I_i^j, $j\in\{1,2,...,q\}$. $\forall n\in P_i^j\cup T_i^j$, superscript n with '$*j$'.
4. $\forall t\in I_i$, $t^{1,2,...v}=t^{*1}$, where $v=R_ie^t_i$, $t^{\mu,\mu+1,...,\mu+Ri}=t^{*h}$, where $\mu=R_i(e^t_i+h-1)+1$, $h\in\{2,3,...,q\}$.
5. create an arc with weight w from the end node of each I_i^j to the start node of $I_i^{(j+1)\%q}$; i.e., $F(t_\alpha^j, p_r^{(j+1)\%q})=w$.
6. add common transitions to T_c; i.e., $T_c=T_c\cup\{t|t\in I_i\cap I_j, \quad i\neq j\}$.
7. } else {
8. $\forall t\in I_i$, $t^{1,2,...Ri}=t$.
9. }
10. }

The time complexity for all I' is $O(n')$.

3) Expand I_i using procedure *expand*(I_i) if loop-combination is involved.

procedure *expand*(I_i)
{
1. $\forall t\in T_I$, if $\exists I_i$, I_j, $I_i\neq I_j$, $(t^{k-1})_i=(t^k)_i$, $(t^{k-1})_j\neq(t^k)_j$, expand $t^{k-1,k}$ in I_i to $t^{k-1}\rightarrow p_t\rightarrow t^k$. $t^{k-1}\rightarrow p_t\rightarrow t^k$ indicates that p_t is an intermediate place between t^{k-1} and t^k and $p_t\in t^{k-1}\bullet\cap\bullet t^k$.
2. Set $M_0((p_r^1)_i)=1$

}

The time complexity for Step 1 & 2 are $O(|T_I|)=O(|T'|)$ and $O(|P'|)$ respectively. Hence, the time complexity for all I' is $=O(|T'|)+O(|P'|)$.

4) Merge all I_i along common transitions using procedure *merge*().
5) Let N' be the resultant new net. Compute the IB for N' with total time complexity $O(n'^3 logn')$.
   a. Splittng N' into a number of lcf-islands as in Step 2.
   b. Compute IB for each lcf-island.
   c. Compute and output the maximum of all IB in Step 6.b as the final IB for N.

The total time complexity in the above steps are $O(n'^2)$ where $n'=|T'|+|P'|+|F'|$, $O(n'^3 logn')$ and $O(n')$ respectively.

procedure *merge*()
1. $\forall t\in T_I$, if t appears in I_a, I_b, …, I_q, merge $(t^k)_a$, $(t^k)_b$, …, $(t^k)_q$ by setting $t'=(t^k)_a=(t^k)_b=\ldots=(t^k)_q$.
2. As a result, $\bullet t'=\bullet(t^k)_a\cup\bullet(t^k)_b\cup\ldots\cup\bullet(t^k)_q$ and $t'\bullet=(t^k)_a\bullet\cup(t^k)_b\bullet\cup\ldots\cup(t^k)_q\bullet$.

The time complexity for Step 1 & 2 are both $O(|T_I|)\bullet O(|P'|)$. Hence, the time complexity for all I' is $=O(|T'|)\bullet O(|P'|)$.Let $R_{max}=\max_{t\in T}R(t)$ be the maximal component in vector $\boldsymbol{R}$.

Now we summarize the algorithm as follows.

Algorithm I for WMG conversion into MG

Input: A WMG

Output: the equivalent MG and the IB

1) Find R_i for each transition.
2) Divide the net into separate loop-combination free islands (lcf-islands) (denoted by I_i).
3) For each I_i, compute $clmf_i$ using the procedure $clmf$ (I_i).
4) Duplicate each I_i $clmf_i$ times using the procedure $duplicate$(I_i).
5) Expand the duplicated I_i using the procedure $expand$(I_i).
6) Merge all I_i along common transitions using procedure $merge$().
7) Output the resulting MG. N'.
8) Compute and output the IB for N'.

The time complexity for Step 1 in the worst case is O(| T | (| T| + |P |)) which is polynomial. The total time complexity is dominated by Step 7 and is $O(n^3)$.

四、參考文獻

[1] E.A. Lee and D.A. Messerschmitt. Static scheduling of synchronous data flow programs for digital signal processing. IEEE Transactions on Computers (TOC), C36 (1):24--35, January 1987.

[2] D. Y. Chao and D. Y. Wang, "Iteration bounds of single-rate data flow graphs for concurrent processing," IEEE Trans. Circuits Syst.-I, vol. CAS40, no. 9, pp. 629--634, Sept. 1993.

[3] K. Ito and K.K. Parhi. Determining the iteration bounds of single-rate and multi-rate data-flow graphs. In IEEE Asia-Pacific Conference on Circuits and Systems, pages 163-- 168, Taipei, December 1994.

[4] R. Schoenen, V. Zivojnovic, and H. Meyr. An upper bound of the throughput of multirate multiprocessor schedules. In Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, volume 1, pages 655--658, 1997.

[5] E. Teruel, P. Chrzastowski, J.M. Colom and M. Silva. On Weighted TSystems. LNCS 616: 348--367, Springer-Verlag, 1992.

[6] D. Y. Chao,"A Fast Implementation For Recurrent DSP Scheduling Using Final Matrix," Journal of Information Science and Engineering, Vol. 16, No.3, 2000, pp. 391-421.

[7] D. Y. Chao,"Conversion, Iteration Bound and X-WINDOW Implementation for Multi-Rate Data Flow Graphs," *Proceedings of the National Science Council, Part A: Physical Science and Engineering,* Vol. 22, No. 3, pp. 362-371, May 1998.

[8] D.Y. Chao, "Performance of Multi-Rate Data Flow Graphs for Concurrent Processing," *Journal of Information Science and Engineering,* Vol. 13, No. 1, March 1997, pp. 85-123.

[9] D. Y. Chao and David Wang, "A Synthesis Technique for General Petri Nets," *J. Systems Integration*, Vol.4, No.1, 1994, pp.67-102.

[10] D. Y. Chao, M. C. Zhou, and D. Wang, "Multiple-weighted marked graphs," 12th World IFAC Congress, Sydney, Australia, Vol.1 pp.259-264, July 1993.

[11] Piotr Chrzastowski-Wachtel, Marek Raczunas: Liveness of Weighted Circuits and the Diophantine Problem of Frobenius. FCT 1993: 171-180.

[12] K. Ito and K. K. Parhi, "*Determining the minimum iteration period of an algorithm*," Journal of VLSI Signal

Processing, vol. 11, pp. 229-244, 1995.

[13] Koh, I., and DiCesare, F., "Transformation Methods for Generalized Petri Nets and Their Applications to Flexible Manufacturing Systems," The 2nd International Conference on Computer Integrated Manufacturing, Troy, NY, MAY 1990, pp. 364- 371.

[14] Lien, Y.L., "Termination properties of generalized Petri nets." *SIAM Journal Of Computers,* Vol. 5, No. 2, 1976.

[15] K. Parhi, "Algorithm transformation techniques for concurrent processors ", In Proc. IEEE, vol. 77, pp.1879-1895, Dec. 1989.

[16] Landwebes, E.H., and Robertson, E.L., "Properties of conflict-free and persistent Petri nets," *Journal Of ACM,* Vol. 25, No 3, 1978.

[17] S. H. Gerez, S. M. Heemstra de Groot, and O. E. Herrmann, "A Polynomial-Time Algorithm for the Computation of the Iteration-Period Bound in Recursive Data-Flow Graphs," IEEE Trans. Circuits Syst.-I, vol. CAS-39, pp. 49-52, Jan. 1992.

附件：封面格式

行政院國家科學委員會補助專題研究計畫成果報告

※※※※※※※※※※※※※※※※※※※※※※※※※※
※ ※
※ MRDFG 的周期界的計算的提升 ※
※ ※
※ ※
※ ※
※※※※※※※※※※※※※※※※※※※※※※※※※※

計畫類別：■個別型計畫 □整合型計畫
計畫編號：NSC 94-2213-E-004-005
執行期間： 94年8 月1 日至 95年7 月31 日

計畫主持人：趙玉 政治大學資管系
共同主持人：
計畫參與人員：

本成果報告包括以下應繳交之附件：

□赴國外出差或研習心得報告一份
□赴大陸地區出差或研習心得報告一份
□出席國際學術會議心得報告及發表之論文各一份

□國際合作研究計畫國外研究報告書一份

執行單位：

中　華　民　國　　94　年　　9　月　　25　日