COMMENTS ON CAPABILITIES, LIMITATIONS AND "CORRECTNESS" OF PETRI NETS*

Tilak Agerwala
Mike Flynn
Electrical Engineering Department
The Johns Hopkins University
Baltimore, Maryland

ABSTRACT

In this paper we examine the capabilities and limitations of Petri nets and investigate techniques for proving their correctness. We define different classes of nets where each is basically a Petri net with slight modifications and study the relationship between the various classes. One particular class appears to be quite powerful, with respect to its capability for representing coordinations. In the second part of the paper we establish the feasibility of using the methods of computational induction and inductive assertions to prove restricted statements about Petri nets.

I. INTRODUCTION

Petri nets are being widely used in the design, specification and evaluation of computer systems [1,7], and in the modeling of production [3] and legal [6] systems. They also appear to be a neat, clear and convenient way to express process coordination. Naturally, the question about capabilities and limitations of these nets arises. It has been shown [4] that there are problems where the desired coordination cannot be expressed using Petri nets. In the first part of this report we introduce different classes of nets. Each class is basically a Petri net with slight modifications. We then examine the relationship between the various classes in the hope that this will give us some insight into the capabilities and limitations of Petri nets.

In the second part we are concerned with proving assertions about Petri nets. Given a coordination problem and a Petri net it should be possible to convince oneself that the Petri net does in fact represent the desired coordination correctly. Techniques for proving any given Petri net correct, will help in proving the correctness of general parallel systems since it may be possible translate the system mechanically into a Petri net where it is easier to see what is going on.

II. CAPABILITIES AND LIMITATIONS

We assume that the reader is familiar with Petri nets and concepts such as liveness, safety, etc. However, for the sake of avoiding ambiguity we will define a Petri net and give the simulation rules explicitly.

A Petri net N is a directed graph defined as a quadruplet (T,P,A,M°) where,

$T = \{t_1, \ldots, t_n\}$ is a finite set of transitions

$P = \{p_1, \ldots, p_m\}$ is a finite set of places

(T, P form the nodes of the graph)

$A = \{a_1, \ldots, a_k\}$ is a finite set of directed arcs of the form (x,y) which either connect a transition to a place or a place to a transition. Each place may have one or more markers in it or it may be empty. A place is full if it has at least one marker.

$$M^\circ = \{(p,n) \mid p \in P \text{ and } n \in \{0,1,2, \ldots\}\}$$

(a function from P to $\{0,1,2, \ldots\}$) is the initial marking.

Simulation Rules

Given a certain marking M of a net, if all the input places to a transition are full the transition is said to be enabled in M. An enabled transition may at some stage decide to fire. At this stage it reserves a marker in each input place and starts firing. At the completion of firing it removes the reserved markers and places a marker in each output place, giving a new marking M'. We say that the firing of t_i in M results in M'. As soon as a marker is reserved it becomes invisible to all other transitions.

$$\bar{t} = t_{b_1}, t_{b_2}, \ldots, t_{b_n} \in T^*$$

is said to be a simulation sequence of a net $N = (T,P,A,M^\circ)$ if there exists a sequence of markings $M^\circ, M^1, \ldots, M^n$ such that t_{b_i} is enabled in M^{i-1} and firing of t_{b_i} in M^{i-1} results in M^i, for all $i \in \{1,2, \ldots, n\}$. The set of all simulation sequences of N is called the simulation set of N or $SIMSET_N$. Let $T' \subseteq T$. Then for each simulation sequence $\bar{t} = t_{b_1}, \ldots, t_{b_n}$ of N we define a reduced simulation sequence $\bar{t}' = t_{c_1}, t_{c_2}, \ldots t_{c_p}$ with respect to T', where $\bar{t}'$ is the sequence that results when all $t_{b_i} \in T - T'$ are excluded from $\bar{t}$. SIMSET $|T'$ is the set of all reduced simulation sequences of N with respect to T'. Two Petri nets $N_1 = (T_1, P_1, A_1, M_1^\circ)$ and $N_2 = (T_2, P_2, A_2, M_2^\circ)$ are said to be strongly equivalent with respect to T if $T \subseteq T_1$, $T \subseteq T_2$ and $SIMSET_{N_1}|T = SIMSET_{N_2}|T$. In this case we write $N_1 \overset{T}{=} N_2$.

So far, we assumed that the transitions of Petri nets had distinct labels. We now define an interpretation I [T',E] of a Petri net $N = (T,P,A,M^\circ)$ as follows:

$T' = \{t_{a_1}, \ldots, t_{a_m}\} \subset T$ is a set of transitions,

$E = \{E_1, \ldots, E_k\}$, $k \leq m$

is a set of event or process names and I: $T' \to E$, i.e. I is a function from T' onto E. Thus, the same event or process name may be attached to different transitions and the same net may represent different coordinations depending on the interpretation given to it. Given a net $N = (T,P,A,M^\circ)$ and an interpretation I [T', E], for each reduced simulation sequence $t_{b_1}, \ldots, t_{b_m}$ with respect to T' we get an <u>interpreted simulation sequence</u> $E_{c_1}, E_{c_2}, \ldots, E_{c_m}$ with respect to I where I $(t_{b_i}) = E_{c_i}$ for $1 \le i \le m$. The set of all interpreted sequences of N with respect to I is called I [SIMSET_N]. A net N_1 with an interpretation I_1 [T', E] is weakly equivalent to a net N_2 with interpretation I_2 [T'',E] if I_1 [SIMSET_{N_1}] = I_2 [SIMSET_{N_2}] and in this case we

$$N_1 \overset{I_1, I_2}{\equiv} N_2$$

In what follows we will define different classes of nets where each kind is basically a Petri net with slight modifications. SIMSET, SIMSET | T and I [SIMSET] can be appropriately defined for each class. If TN and TN_x refer to two different classes of nets, then PN and PN_x refer to all the coordinations representable by TN and TN_x respectively. We say that $PN \subseteq PN_x$ if for every N ε TN and interpretation I [T',E] there exists an N_x ε TN_x and interpretation I_x [T'',E] such that

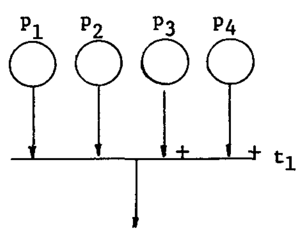
$$N \overset{I, I_x}{\equiv} N_x.$$

Thus $PN \subset PN_x$ if $PN \subseteq PN_x$ and there exists a net N_x ε TN_x and an interpretation I_x [T',E] such that there is no net N ε TN and interpretation I [T'',E] with

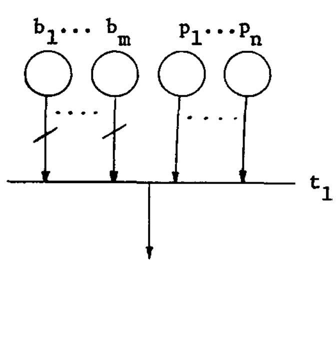
$$N \overset{I, I_x}{\equiv} N_x.$$

<u>Classes of Nets</u>

1. Let the class of ordinary Petri nets be TN.
2. The transitions in ordinary Petri nets are enabled only when <u>all</u> the input places are full and we can consider these transitions to have an AND-input logic. If in addition, we allow transitions with OR input logic, we call the class of nets TN_{log}



(letting P_i denote the number of markers in p_i) t_1 is enabled if and only if $[(P_1 > 0) \wedge (P_2 > 0)] \wedge [(P_3 > 0) \vee (P_4 > 0)]$. Thus t_1 is enabled even if all the input places do not have markers and when it starts firing it reserves a marker in each input place that has at least one.
3. In addition to the ordinary transitions in the nets belonging to TN we allow a transition to have input places and arcs of a special kind. The transitions allowed are of the form:



t_1 is enabled if and only if $(B_1 = 0) \wedge (B_2 = 0) \wedge \ldots (B_n = 0) \wedge (P_1 > 0) \wedge (P_2 > 0) \wedge \ldots \wedge (P_n > 0)$. When t_1 starts firing a marker is reserved in each of $P_1, P_2, P_3, \ldots, P_n$. Let the class of nets be called TN_{com}.
4. In addition to the ordinary places in the nets belonging to TN we introduce a special place , (say p_1). A transition will place a stone in p_1 if and only if $P_1 = 0$. Let the class of nets be $TN_{\overline{out}}$.

<u>Results</u>

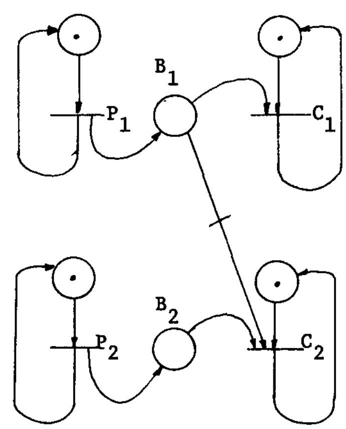
1. Since in each case we provided the nets with additional capabilities over the nets belonging to TN, obviously:

$$PN \subseteq PN_{log}$$
$$PN \subseteq PN_{\overline{out}}$$
$$PN \subseteq PN_{com}$$

2. $PN \subset PN_{com}$

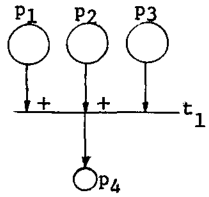
<u>Proof</u>

Kosaraju [4] describes a coordination problem and proves that it falls outside PN. The problem is as follows: There are four cyclic processes, P_1, P_2, C_1 and C_2 and two buffers B_1 and B_2. P_1 and P_2 are producers which place one item each on top of B_1 and B_2 respectively in every cycle. C_1 and C_2 consume one item each from the bottom of B_1 and B_2 respectively. However, C_1 has higher priority than C_2 so that C_2 can consume only if B_1 is empty. To prove that $PN \subset PN_{com}$ we will give an interpreted net belonging to TN_{com} which represents the desired coordination. The net is:

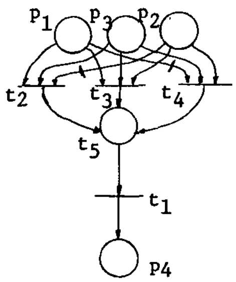


3a. $PN_{log} \subseteq PN_{com}$

For every net N_{log} = (T,P,A,M°) there exists a net net N_{com}=(T', P', A', M_1°) ε TN_{com} such that $T \subset T'$ and $N_{log} \overset{T}{\equiv} N_{com}$. The result 3a follows from this. We will not go into the details of a proof but will illustrate the idea by means of an example. Let the net N below be part of a larger net N_{log} = (T,P,A,M°) belonging to TN_{log}.



N can be replaced by:



Let the resulting net be N'. Then obviously $N' \overset{T}{\equiv} N_{log}$.

By applying a similar procedure to each transition with OR input logic we end up with a net N_{com} which is strongly equivalent to N_{log} with respect to T.

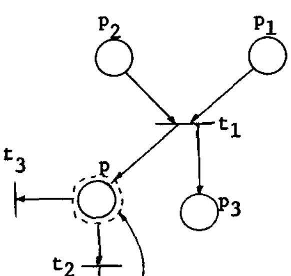
3b. Kosaraju's problem 1 and proof [4] can be used to prove that $PN_{log} \subset PN_{com}$.

4a. $PN_{\overline{out}} = PN$

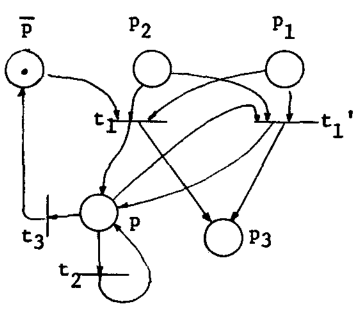
For every net $N_{\overline{out}} = (T_1, P_1, A_1, M_1°) \in TN_{\overline{out}}$ and interpretation $I_1[T,E]$, there exists a net $N = (T_2, P_2, A_2, M_2°) \in TN$ and interpretation $I_2[T'',E]$, such that

$$N_{\overline{out}} \overset{I_1,I_2}{\equiv} N.$$

Again, we will not go into details of a proof but will illustrate with an example. Let the net N below be part of a larger net $N_{\overline{out}}$.



Then N can be replaced by:

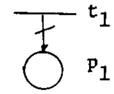


resulting in the net N'. Here we have introduced a place $\overline{p}$ which is a complement of p in the sense that $\overline{p}$ has a marker if and only if p does not. The reader can convince himself that under the interpretation I' $[T \cup \{t_1'\}, E]$, where $I'(t) = I_1(t)$ for $t \in T$ and $I_2(t_1') = I_1(t)$, $N_{\overline{out}} \overset{I_1,I'}{\equiv} N'$. Continuing this process until all places of the form are eliminated, we end up with a net $N \in TN$ and an interpretation I_2 $[T'', E]$ such that

$$N_{\overline{out}} \overset{I_1,I_2}{\equiv} N.$$

This shows that $PN_{\overline{out}} \subseteq PN$ and from result 1 we conclude that $PN_{\overline{out}} = PN$. Since $PN \subset PN_{com}$ we also conclude that $PN_{\overline{out}} \subset PN_{com}$.

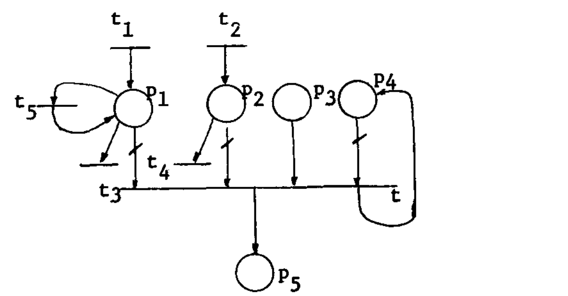
Comments: We feel that $PN \subset PN_{log}$. We are also examining other classes of nets. For example, in addition to the ordinary arcs between transitions and places we allow the following:



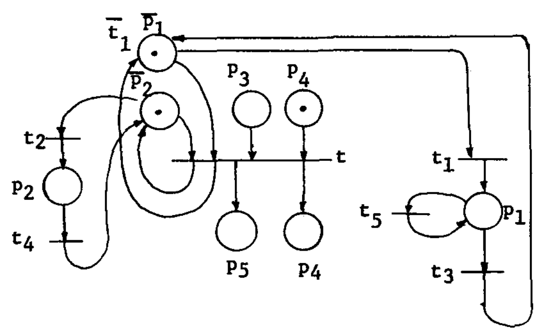
t_1 will place a marker in p_1 if and only if $P_1 > 0$. Another class of nets is those where we allow a transition to nondeterministically place a marker in one or more of its output places. The results obtained so far indicate that TN_{com} is a very powerful class of nets.

Safe nets

If one considers only safe nets (where each place can contain at most one marker at any stage), then it can be shown that for every $N_{com} = (T,P,A,M°) \in TN_{com}$ that is safe, there exists a safe net $N \in TN$ such that $N_{com} \overset{T}{\equiv} N$. Again, we will only demonstrate the technique of obtaining N with the help of an example. Let the net N_1 below be part of a safe member N_{com} of TN_{com}.



Replace N_1 by the net below to get a net N'



The fact that N_{com} is safe permits us to introduce places $\overline{p}_1$, $\overline{p}_2$, $\overline{p}_4$ which are complements of the places p_1, p_2 and p_4 respectively. I.e. $\overline{p}_1$ has a marker if and only if p_1 does not. Every transition that causes a marker to be put in p_1 should cause a marker to be removed from $\overline{p}_1$. Every transition that causes a marker to be removed from p_1 should cause a marker to be placed in $\overline{p}_1$. We now have

$$N' \overset{T}{\equiv} N_{com}.$$

By continuing the process of replacement we end up with a net $N_k \in TN$ such that $N_k \stackrel{T}{=} N_{com}$. If $PN_x|safe$ denotes the set of coordinations representable by safe members of TN_x, then $PN|_{safe} = PN_{com}|safe$. From results 3 and 4 $PN \mid safe = PN_{log} \mid safe = PN_{\overline{out}} \mid safe = PN_{com} \mid safe$.

Thus, even though TN_{com} is a powerful class of nets, in practice one would probably be more concerned with safe nets and here the modifications made to ordinary Petri nets do not increase the overall power.

III. CORRECTNESS

When we say that a "Petri net N is correct", intuitively what is meant is that the Petri net does what the designer intended it to do. Given a particular problem, a Petri net is constructed which represents the desired coordination. First and foremost we are not at all concerned with whether the Petri net is the best one for the given problem. In fact, we will not even try to prove that the Petri net effectively represents the desired coordination. We shall, however, try to prove very restricted statements about a net which are provided by the designer. The kinds of statements we will attempt to prove are:

1. At any given time only one of the transitions from the set $\{t_1, \ldots, t_k\}$ may be firing.
2. Two given transitions will never conflict.
3. A given place is safe with respect to a particular marking or a given marking is safe.
4. A given place can contain at most N markers
5. A given transition is live.
6. A given marking is reachable from another.
7. A given transition has fired at most x times.
8. In general it may be very difficult to show that a "net is deadlock free". Again, the designer will have to provide statements, for example, "Every transition in cycle C is live at every stage", from which he can reasonably conclude that the net will not hang up.

In the following we present two methods to prove the correctness of Petri nets: Computational induction and inductive assertions.

Computational Induction

Here we develop certain relations that remain invariant during the simulation of a net. By using these relations suitably we will be able to prove certain properties about the net. According to our simulation rules, when a transition starts firing it reserves a marker in each input place. Reserved markers are invisible to all other transitions. However, in the invariant relations, all reserved markers are also counted and assumed to be in their current places. The relations follow trivially from the simulation rules. Let,

Mi: Number of stores in p_i initially
Pi: Number of stores in p_i at any instant
Ti: Number of times t_i has fired till any instant.

Relation 1: Let I_i = {set of transitions with p_i as output place}, O_i = {set of transitions with p_i as input place}, then

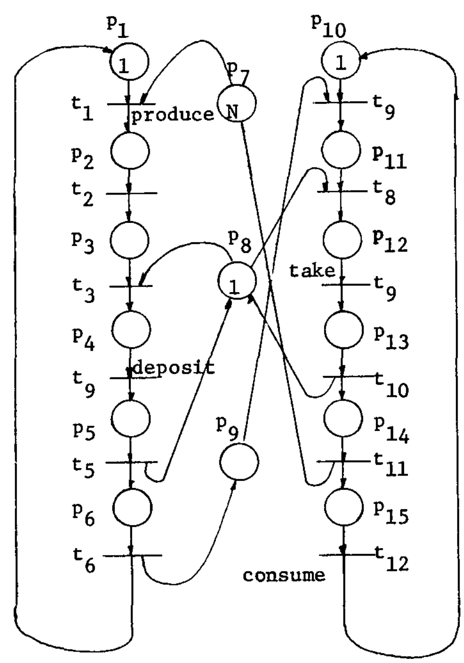
$$P_i = \sum_{t_k \epsilon I_i} T_k - \sum_{t_j \epsilon O_i} T_j + M_i \geq 0$$

Relation 2: Let $t_{a_1}\ p_{b_1}\ t_{a_2}, \ldots, p_{b_{k-1}}, t_{a_k}$ be a path in the net such that t_{a_i}, p_{b_i} $1 \leq i \leq k$ are distinct. If in addition $I_{b_i} = \{t_{a_i}\}$, $1 \leq i \leq k-1$ then we have a simple path and $T_{a_k} \leq T_{a_1} + \sum_{i=1}^{k-1} M_{b_i}$.

Relation 3: If S_1 is a simple path from t_i to t_j and S_2 is a simple path from t_j to t_i then $S_1\ S_2$ forms a simple cycle. If in addition every place on a simple cycle has only one input and one output arc then we have a pure cycle. Let S be a pure cycle then:

$$\sum_{p_i \text{ in } S} P_i = \sum_{p_i \text{ in } S} M_i = N_s \quad \text{(say).}$$

We have used these relations to prove simple assertions about nets, and will illustrate the method by means of an example. Consider the producer consumer problem with bounded buffer. The producer places items in a buffer. (length N) and the consumer consumes them. The problem is to coordinate these two essentially independent processes so that the consumer does not try to take an item from the buffer when it is empty and the producer does not place an item where the buffer is full. The Petri net that represents the described coordination is given below: (the numbers in the places denote the initial number of markers)



We are interested in proving the following properties for this net:

1. t_4 and t_9 cannot be firing at the same time, i.e. the producer and consumer do not try to access the buffer at the same time.
2. $0 \leq T_4 - T_9 \leq N$. I.e. there is no buffer overflow or underflow.
3. the net is deadlock free.

Proof 1:

$T_8 + T_3 \leq 1 + T_{10} + T_5$	(1) By R_1
$P_4 = T_3 - T_4$	(2) By R_1
$T_5 \leq T_4$	(3) By R_2
$P_4 \leq T_3 - T_5$	(4) from (2) + (3)

Similarly, $P_{12} \leq T_8 - T_{10}$ (5)
Therefore, $P_4 + P_{12} \leq 1$ (6) from (4), (5), (1)
From 6 and the simulation rules we conclude directly that T_4 amd T_9 cannot be firing at the same time.

Proof 2.

$T_4 \leq T_9 + N$ (1) By R_2

Therefore, $T_4 - T_9 \leq N$
i.e., the number of deposits - the number of removals $\leq$ N. Therefore, three can be no buffer overflow.

$T_9 \leq T_4$ (2) By R_2

Therefore, $T_4 - T_9 \geq 0$
i.e., number of deposits - number of removals _ 0.
Therefore there can be no buffer underflow.

Proof 3.

For this particular problem it is easy to see that deadlock can occur only if $P_7 = P_9 = 0$ and there is no way to change this situation. (pure cycles can be represented by the subscripts of the places only since there is no ambiguity)

S_1 = 1,2,3,4,5,6,7,1 is a pure cycle
S_2 = 10, 11, 12, 13, 14, 15, 10 is a pure cycle
S_3 = 3, 4, 5, 6, 9, 11, 12, 13, 14, 7, 3 is a pure cycle
$N_{S_1} = 1$ (1)
$N_{S_2} = 1$ (2)
$N_{S_3} = N$ (3)
$a = P_3 + P_4 + P_5 + P_6 \leq 1$ (4) from (1)
$b = P_{11} + P_{12} + P_{13} + P_{14} \leq 1$ (5) from (2)

Therefore, $a + b \leq 2$ from (4), (5)
But if N>2 and $P_7 = P_9 = 0$ then
$a + b = N > 2$ from (3)

Therefore, we get a contradiction. Thus, for N > 2, at no stage can both P_7 and P_9 be zero. Therefore, there can be no deadlock for N>2. For N = 1 and N = 2 separate arguments can be given to prove that the net is deadlock free.

Inductive Assertions

This method was introduced by Floyd [2] to prove the correctness of sequential programs and the same technique was used by Lauer [5] for proving parallel programs correct. We have taken the basic ideas from [5] and modified them to be applicable in the framework of Petri nets. Here again, our aim is to prove that a Petri net is correct with respect to a particular given assertion A. The procedure is as follows: with each transition in the net we associate an assertion. Our aim is to prove that every time a transition is enabled, the corresponding assertion is true irrespective of the particular simulation which caused this transition to be enabled and irrespective of the state of the rest of the net. Once this has been established, the truth of A has to be deduced from the assertions at the transitions.

Let $N = (T,P,A,M°)$ be a Petri net. An assertion a_i asserted with a transition $t_i \in T$ is a predicate on the values of P_k and T_k where $p_k \in P$ and $t_k \in T$. The Petri net is correct with respect to the assertion a_i if and only if for each simulation of the net that enables t_i, a_i is true when t_i is enabled. The net N is correct with respect to a set of assertions if and only if it is correct with respect to each assertion in the set. Let I_i = set of input places of t_i and O_i = the set of output places. Then we have the following:

Induction Theorem

To prove that a Petri net $N = (T,P,A,M°)$ is correct with respect to a set of assertions $\{a_i | t_i \in T\}$ it is sufficient to prove the following:

(1) a_i is true for all t_i that are enabled in M°.

(2) For each $t_i \in T$, let $P_i = \{p | p \in I_i \wedge (p,0) \in M°\}$ i.e., the set of all initially unmarked input places of t_i. Let $P_i = \{q_1, q_2, \ldots, q_n\}$. Let $T_j = \{t_k | q_j \in O_k\}$, $1 \leq j \leq n$, i.e., the set of all transitions of which q_j an output place. Let $B_i = \{(b_1, b_2, \ldots, b_n) | t_{b_j} \in T_j\}$ Each n-tuple in B_i gives the set of transitions which when fired cause markers to be placed in the initially unmarked input places of t_i. Let Fire $(b_1, \ldots, b_n)$ denote the fact that the transitions $t_{b_1}, \ldots, t_{b_n}$ fire. Then for each $t_i \in T$,

$$a_{b_1} \wedge a_{b_2} \wedge \ldots \wedge a_{b_n} \wedge \text{Fire}\ (b_1, \ldots, b_n) \Rightarrow a_i$$

for all $(b_1, b_2, \ldots, b_n) \in B_i$ .... (1)

Proof: Obvious

Each equation of the form (1) is called a verification condition. It should be clear to the reader that the verification conditions are really very strong. Thus the conditions are not necessary but only sufficient.

To prove a net correct, one may often have to construct an augmented net. Let $N_1 = (T_1, P_1, A_1, M_1°)$ be a Petri Net. Then $N_2 = (T_2, P_2, A_2, M_2°)$ is an augmentation of N_1 if and only if $T_1 \subseteq T_2$, $P \subseteq P_2$, $A_1 \subseteq A_2$, $M_1° \subseteq M_2°$ and

$$N_1 \overset{T_1}{\equiv} N_2.$$

One can show that, if N_2 is an augmentation of N_1 then N_2 is correct with respect to a_i where $t_i \in T_1$ if and only if N_1 is correct with respect to a_i'. Here a_i' is the same as a_i with all references to $t \in T_2 - T_1$ and $p \in P_2 - P_1$ deleted.

Thus, to prove that a Petri net $N = (T,P,A,M°)$ is correct with respect to an assertion A one goes through the following steps:

1. Formulate the assertion a_i for each transition t_i.
2. Prove that all assertions associated with transitions that are initially enabled are true.
3. Prove that all the pertinent verification conditions hold and conclude that N is correct with respect to $\{a_i | t_i \in T\}$. (Instead of (2) and (3) one may construct an augmented net N' of N, associate appropriate assertions with the transitions of N', carry out (2) and (3) for N' and conclude that N is correct with respect to $\{a_i | t_i \in T\}$)
4. Deduce that the net operates correctly with respect to the main overall assertion, A.

We have used this method to prove the correctness of a Petri net representation of the producer - consumer problem with respect to an overall assertion. Since the assertions and proof are essentially similar to those of Lauer [5] we will not present the example here. In a subsequent report we will present weaker verification conditions, examine whether it is necessary to associate assertions with each and every transition and develop "local" conditions under which places, arcs and transitions can be added to a net N resulting in an augmented net N'.

IV. CONCLUSIONS

We hope that the discussion in Part II sheds some light on the capabilities and limitations of Petri nets. TN_{com} seems to be a powerful class of nets. It is possible that these nets do provide a correct, formal counterpart to the vague notion of a "coordination problem". We will examine this aspect in another report. Also, Petri nets seem to be sufficiently powerful if one is concerned only with safe nets. This may very well be the case in practice.

In part III we have established the feasibility of using the methods of computational induction and inductive assertions to prove restricted kinds of statements about Petri nets. Ultimately, work in this direction will facilitate the process of convincing oneself that a general concurrent system is correctly coordinated.

V. REFERENCES

1. Dennis, Jack, B. "Modular, Asynchronous Control Structures for a High Performance Processor", Record of the Project MAC Conference on Concurrent Systems and Parallel Computation, ACM, New York, 1970, pp. 55-80.

2. Floyd, R. W. "Assigning Meanings to Programs", Proceedings of a Symposium in Applied Mathematics, Vol.19, Mathematical Aspects of Computer Science, American Mathematical Society, 1967, pp. 19-32.

3. Hack, Michel "Analysis of Production Schemata by Petri nets", M.S. Th., Dept. of Electrical Engineering, MIT, Cambridge, Mass., February 1972.

4. Kosaraju, S. Rao "Limitations of Dijkstra's Semaphore Primitives and Petri Nets", Hopkins Computer Research Rpt. #25, Research Program in Computer Systems Architecture, J.H.U., Balto., Md., May 1973.

5. Lauer, Hugh Conrad "Correctness in Operating Systems", Ph.D. Thesis, Carnegie-Mellon University, September 1972. AFOSR -TR- 72 - 2361, Contract F 44620 - 70 - C - 0107.

6. Meldman, Jeffery and Holt, Anatol "Petri Nets and Legal Systems", Jurimetrics Journal 12, December 1971, pp. 65-75

7. Noe, Jerre D., "A Petri Net Model of the CDC 6400" Proceedings of the ACM/SIGOPS Workshop on Systems Performance Evaluation, April 1971, pp. 362-378.

ACKNOWLEDGEMENT

I am grateful to my advisor, Professor Michael J. Flynn, for his continued help, guidance, encouragement and support. I would also like to thank my colleague, Joe Davison, for his helpful comments.