Computation Structures Group Memo 68

Petri Nets and Languages

Henry G. Baker

May 1972

This work was done at Project MAC, M.I.T., and was supported in part by the Advanced Research Projects Agency, Department of Defense, under Office of Naval Research Contract Nonr N00014-70-A-0362-0001.

Petri Nets have provided the means to describe complex asynchronous systems such as production facilities and computer central processing unit control. Since this model provides a relatively natural way to describe asynchronous events, as opposed to more traditional models such as finite state machines or Turing machines, it is inevitable that a lot of research effort has been aimed at understanding these systems. A. Holt, Commoner, and Genrich among others have been the prime movers in this area. Furthermore, due to the close relationship of Petri Nets and mathematical systems known as vector addition systems, others such as Rabin, Karp, and Miller have indirectly contributed.

Since general Petri Nets are so powerful, they are also hard to analyze. Therefore, several subclasses of Petri Nets have been identified and studied. In order of decreasing power, these are *Simple* nets, *Free Choice* nets, *Persistent* nets, *Marked Graphs*, and *State Machines* (not necessarily strict inclusion at each level).

Although it seems intuitively obvious that Petri Nets should be better to describe certain systems and events, it would be nice to know how these nets fit into the more classical systems of automata: finite state machines, stack machines, Turing machines, counter machines, etc. With each one of the classical machines has been identified the "languages" accepted or generated by these machines. Therefore, to compare Petri Nets to these machines, we either have to show a way of simulating one of the more classical machines, or else recognizing or accepting a particular class of languages.

Petri nets, you will recall, are labelled, bipartite directed graphs, where the nodes of one partition are called "places" and of the other partition are called "transitions". Furthermore, associated with the net is a function from the places to the non-negative integers called the "initial marking" which intuitively means the "number of tokens" occupying each place". Each transition, when it "fires", absorbs a token (from its "input places") through each of its input arcs, and places a token on each of its output arcs which then "transmit" them to the places at the head of each arc. Obviously, a transition whose input places do not contain enough tokens cannot fire, and is therefore said to be "disabled". Conversely, the transition is "enabled". Since we have not ruled out the possibility that the set of input and the set of output places are not disjoint, a transition may absorb a token from a place as it fires and put it right back again. If the input and output places of a transition are disjoint and parallel arcs are allowed, Petri nets of this definition are isomorphic to *vector addition systems*, which we describe next.

Vector addition systems consist of a "space" N^p, p a positive integer, a set of vectors $T \subseteq Z^p$, and an origin $I \in N^p$, where N is the set of non-negative integers and Z is the set of integers. The "reachability set" of a vector addition system (T,I) denoted by $RS(T,I) \subseteq N^p$ is the set of all points constructed recursively by

1) $I \in RS(T,I)$ and
2) $P \in RS(T,I)$ if and only if $Q \in RS(T,I)$ and $P=Q+t$ where $t \in T$, $P \in N^p$.

Intuitively, $P \in RS(T,I)$ implies that P can be reached from I by a path of vectors from T *which never leaves the first quadrant* (N^p). Thus, in the isomorphism of Petri nets to V.A. systems, the transitions correspond to vectors, the initial marking to the "origin" vector, the forward marking class to the reachability set, and the places to the dimensions of the space. Due to this isomorphism, I will use terms from both models interchangeably.

A question that might come up in regard to Petri Nets is: given two different Petri nets with the same number of places and a 1-1 correspondence of the places, is the marking class of the first equal to the marking class of the second? In several cases the answer is easy to compute. If the vectors are all positive, i.e., $t \in T$ such that $t \geq 0$, the answer is easy. Furthermore, linear algebra tells us that if all of the vectors are linearly independent, the answer is also easy. If the system is finite, the answer may not be easy, but can obviously be computed. However, Rabin has shown this question to be undecidable, in general In fact, another similar question, which appears easier, is also conjectured to be undecidable: given a point $P \in N^p$, is $P \in RS(T,I)$? Karp and Miller have shown a weaker question to be decidable: given a point $P \in N^p$, is there a point $Q \in RS(T,I)$ such that $Q \geq P$?

Thus, although some results are known about Petri nets in their full generality, one may question the appropriateness of this attack. Although it would be nice to know that a particular Petri net was free from deadlock, we know that most Petri nets are subject to this phenomenon and the question is not that it is possible, but how to avoid it. Furthermore, it is still not easy to see how reachability sets can tell us about simulating Turing machines or generating languages.

Jack Dennis has suggested a different tack. From an initial marking, there exists a set of legal firing sequences, i.e., strings of symbols from $\{t|t \in T\}^*$, where the position in the sequence of a particular letter t tells when the transition named t fired. This is an obvious hint to consider the language of legal firing sequences for a net and to try to characterize the set of languages thus generated by all Petri nets.

There are several observations that follow immediately. Any prefix of a legal sequence is also a legal sequence. Therefore, if we were to consider a generalized possibly infinite "state machine" accepting this language, all states would be accepting except for a single non-accepting "dead" state. Thus, if one symbol in the sequence was illegal, the whole string would be illegal. Continuing with this model, we notice that the points of the reachability set of this system are isomorphic to the accepting states of the state machine for the language. Furthermore, the transitions enabled at a particular point can be identified with the labels on the arcs from the corresponding state in the state machine to another accepting state. This isomorphism will be extremely useful in determining the characteristics of languages "generated" by Petri nets.

Construct a labelled directed graph called a "reachability graph" from the points in the reachability set such that the points in this set are also the points in the graph, and the legal transition vectors will be the arcs in this graph. That is, if $P,Q \in RS(T,I)$, and $P+t=Q$, then we include a directed edge from P to Q labelled with "t" in this "reachability graph" we are constructing. The reachability graph, treated as a nondeterministic state machine, accepts only and all those legal firing sequences for the net. The reachability graph is deterministic if we add the constraint that if no letter on the outgoing transitions matches the letter input or generated, the machine transfers to the "dead" state which does not appear in the reachability graph.

Consider a point P on a directed circuit of the reachability graph. The existence of this circuit implies the existence of a firing sequence in the original net which, when started with the initial marking P, returns the net to the marking P. Since the initial marking is the same as the final marking, it must be that the sum of the vectors in the firing sequence is zero. Furthermore, if we sum the vectors around an undirected cycle--with the convention that we will subtract instead of add vectors contrary to the direction of summation--it should be clear that this sum is also equal to zero. Thus, the cycles of the reachability graph of a Petri net satisfy Kirchoff's voltage law.

In any graph, directed or undirected, the set of edge-disjoint unions of cycles form a vector space. The reachability graph of a Petri net is no different and therefore we investigate a *basis* of this space. The size of the basis of the cycle space (of a connected graph) is equal to the number of edges minus the number of vertices, plus one. Thus, if we were to consider all of the equations from Kirchoff's voltage law for a reachability graph, this would be the maximum number of independent equations. In practise, however, many of these equations are linearly dependent. For example, if the Petri net under consideration is a Marked Graph, only one equation holds, i.e., $\sum_{t \in T} t = 0$. This implies that the only way to leave a particular point in p-space and return through a legal set of firings is to fire all transitions exactly once. Furthermore, I conjecture that a Petri net is *persistent* if and only if the set of Kirchoff equations has rank=1. It follows that the coefficients in this equation are precisely those which appear in Patil's counting theorem.

We are now ready to consider an interesting problem. We will consider various types of equivalence which might be defined for Petri nets and investigate the properties of these definitions. As we quoted earlier, Rabin has considered equivalence in the sense of equality of reachability sets and shown this type of equivalence to be undecidable.

Another type of equivalence might be the equality of the set of legal firing sequences. To me, this seems to be a more natural type of equivalence in the sense that the nets are specified more by what they do than by how they achieve it. For nets with bounded markings, i.e., those with a finite reachability graph, the problem is trivial. Since we already have our finite state machine for the net language, reduce the machine and compare it to the reduced machine for another net and see if they are equal. We already know from automata theory that the reduced deterministic finite state machine is a canonic form for finite state languages. However, this approach does not seem to give us any insight into the various kinds of Petri nets with the same firing sequences.

A model that would be extremely useful would be a canonic form for Petri nets in the sense that any two nets with the same firing sequences would have the same canonic form. This canonic form might lead to some minimal Petri nets with the same "behavior".

However, the problem for general Petri nets did not seem to have an obvious method of attack, so Professor Dennis suggested the restriction of the problem to marked graphs. This approach proved fruitful, and a canonic form for live marked graphs is presented in the next section.

Before the presentation of the canonic form, I will reiterate the definition of marked graphs.

Definition. A *Marked Graph* is a Petri net in which every place has exactly one input arc and one output arc. Since the places have one arc input and one output, we will omit the drawing of the place and regard the tokens as residing on one long arc between the place's input transition and output transition. Furthermore, we will draw the transitions as points rather than lines. Thus, marked graphs are labelled directed pseudo-multigraphs in which various numbers of identical "tokens" are associated with the arcs.

There are exactly two simple rules which, when applied, reduce a live marked graph to canonic form.
1) Eliminate marked self-loops.
2) Consider each arc x from a to b in the marked graph. Consider all directed paths from a to b not including x. If the *token length* of x is strictly less than the token length of all other paths, keep x in the marked graph, else eliminate x along with its marking. (The *token length* of a path is the number of tokens on that path.)

These two rules imply that in a canonic form marked graph, token length satisfies the triangle inequality -- $TL(ac) \leq TL(ab)+TL(bc)$. The proof of this canonic form is given in a paper "A Canonic Form for Marked Graphs" by this author, included as an Appendix.

A further result is the proof that marked graphs are closed under a type of homomorphism called "partial erasing identity homomorphisms". These homomorphisms either erase all of a particular symbol in a firing sequence or preserve it untouched. This is equivalent to classifying certain transitions in a Petri net as "interesting" or not "interesting" can considering firing sequences of interesting transitions. A corollary to this result shows, however, that deleting non-interesting vertices in a safe marked graph may reduce it to an unsafe marked graph. (A *safe* Petri net is one in which no reachable marking assigns more than one token to any place.)

I will be working on extending these types of results to other types of Petri nets such as "state machines", free choice nets, etc. The hope in this work is to understand the capabilities of Petri nets better.

Appendix. "A Canonic Form for Marked Graphs"

In this paper I will describe and prove a canonical form for marked graphs; i.e., a form which preserves "behavior", suitably defined. Furthermore, I will show how this canonic form can be extended to subsets of the vertices of a graph.

As you will recall, marked graphs consist of a set of vertices (or transitions), a set of directed arcs between various vertices, and an "initial marking" which assigns a (possibly different) non-negative integral number indistinguishable "tokens" to each arc. That is, a marked graph consists of a labelled, directed pseudo-multigraph with the arcs initialized with various numbers of tokens. Finally, there is a "firing rule" which allows the assignment of tokens to change. Thus, if a vertex v in a marked graph *fires*, it absorbs one token from each incoming arc and adds one token to each output arc. Obviously, if there is an incoming arc with no token, v is said to be *not enabled* or *disabled*. A *firing sequence* is a time history of vertex firings in a marked graph from some initial marking arranged in a string σ such that the ith symbol σ_i is the label of the transition that fired at time t=i. A *firing vector* Σ is a vector of non-negative integers which describes a firing sequence where Σ_i is the number of times transition i appears in the sequence. Finally, a *live* marked graph is one in which every vertex can be made to fire an arbitrarily large number of times.

Since at any point in the history of a marked graph many different vertices are enabled, the choice as to which to fire next is non-deterministic. Therefore, there exists a set of many different firing sequences for each marked graph. Since this set seems to describe completely the "behavior" of the graph, we will be looking for a canonic form which preserves this set of sequences.

Suhas Patil has suggested that marked graphs can only sequence transitions and that if transition t is to be constrained to fire after transition s, there must be a directed path from s to t. I have firmed up this notion into the canonic form exhibited here.

Definition. The *minimal-arc form* (*m.a.-form*) for a live marked graph is obtained by applying the following rules repeatedly to the graph until neither rule applies.

1) Eliminate marked self-loops along with their markings.

2) Consider an arc x from a to be in the graph. Consider all directed paths from a to b not including x. If the token length of x is strictly less than the token length of all other paths, keep x else eliminate x along with its marking from the graph. (The token length of a path is the number of tokens on that path.)

Theorem I. The minimal-arc form for a marked graph is canonic--i.e., it satisfies these criteria:

a. the m.a.-form of a graph G has the same set of firing sequences as G.

b. if two different graphs G, G' (with the same vertex set) have the same set of firing sequences, then they also have the same canonic form.

Proof: We will first show that rules 1) and 2) for the m.a.-form satisfy criterion a when applied one at a time. By then proving criterion b, we will show that the order of application of rules 1) and 2) do not affect the derived m.a.-form.

Consider the effect of rule 1) on a marked graph ***M*** which has a self-loop x on vertex v. That is, consider another marked graph ***M'*** = ***M***-x. We know that in marked graphs the number of tokens around a circuit is constant and the self-loop is no exception. Therefore, the loop is always marked. Consider the set X of input arcs to v in ***M'*** where |X|=m. The criterion that v is enabled in ***M*** can be expressed as

$$\left[\bigwedge_{i=1}^{m}(M(x_i)>0)\right]\wedge M(x)>0 \equiv \bigwedge_{i=1}^{m}(M(x_i)>0)$$

But the right hand side is just the criterion that v is enabled in ***M'***. Therefore, the elimination of x cannot affect the enabling of v either way. But since x is an incoming arc for only v, neither can it affect the enabling of any other vertex. Therefore, ***M*** and ***M'*** exhibit the same firing sequences.

To prove rule 2), again consider two marked graphs ***M***, ***M'*** where ***M'***=***M***-x due to rule 2). Thus x is an arc from a to b and there exists a directed path P such that the token length of P, L(P), is less than or equal to the token length of x, L(x)=M(x). Again we know that in a marked graph, the token count of a (non-directed) cycle is constant, providing that we subtract rather than add tokens on arcs whose direction is opposite to the direction of the cycle. Since P∪x is such a cycle, we know that L(x)-L(P)=c≥0. This implies that even if we were able to exhaust the tokens P through firings of ***M***, we would still have a residue of c tokens on x. Thus, if c>0, these are extra tokens and can be thrown away without affecting the firings of ***M***. Now we have L(P)=L(x). Since x is input to vertex b, the tokens on x can only directly affect the firing of b. The enabling equation for b is

$$\left[\bigwedge_{i=1}^{m-2}(M(x_i)>0)\right]\wedge M(x)>0 \wedge M(y)>0$$

where y is the last arc of P. But M(y)≤M(x). Therefore the enabling equation for b can be simplified to

$$\left[\bigwedge_{i=1}^{m-2}(M(x_i)>0)\right]\wedge M(y)>0$$

Since the elimination of x cannot affect the enabling (and hence firing) of b, it cannot affect the behavior of the net. Therefore ***M*** and ***M'*** have the same set of firing sequences.

To complete the proof, we must now show that given two m.a.-form marked graphs ***M***,***M'*** over the same set of vertices, if the sets of firing sequences are equal, the graphs must be the same. Therefore, we will exhibit a simulation of ***M*** and ***M'*** simultaneously which proves that ***M*** and ***M'*** are really the same.

Assuming ***M*** and ***M'*** are really different, either they differ in at least one arc, or they differ in the initial marking, or both. Assume that ***M*** has an arc x from a to b that ***M'*** does not have. Furthermore, either x is marked or x is not marked. If x is marked, either there exists a firing sequence which will exhaust x of tokens or there does not. Assume that x can be cleared of tokens by some firing sequence (applied to both ***M*** and ***M'***). Then before b can fire again, a must fire to provide x with a token. But this implies the existence of a blank path from a to b in ***M'***. For if all paths from a to b were marked, b could fire without firing a (since ***M'*** is live), which is a contradiction.

With x blank in ***M***, fire every vertex with the exception of a until no more vertices in a's component can fire (except a). This firing cannot continue forever due to the persistence of marked graphs. Now fire a. Either b is enabled or b is not enabled. Assume that b is not enabled. This implies the existence of a blank path from a to be other than x before we fired a. But a blank path would violate our canonic rule number 2). Therefore b must be enabled. But b cannot be enabled in ***M'*** because the token has not advanced far enough along the path from a to b. Therefore ***M*** and ***M'*** must both have the arc x. By similar reasoning, all the arcs of ***M*** and ***M'*** are the same.

We now will prove that the markings are the same, too. Assume that M(x) in ***M*** and M'(x) in ***M'*** such that M(x)>M'(x). Obviously, if we simulate ***M***, ***M'*** simultaneously with the same firing sequence, the discrepancy M(x)-M'(x)=c will persist. Without firing a, fire every vertex until no vertex (besides a) in a's component can fire. Either x is empty in ***M'***, or else there exists a blank path from a to b (which violates canonic form). Therefore x is empty in ***M'***. However, there are c tokens left on x in ***M***. Again, we can fire b in ***M*** to get rid of those tokens, but this would differ from ***M'***. Therefore, there must be some blank path between a and b in ***M*** which disables b. But a blank path would violate canonic form in ***M***. Therefore, the markings of ***M*** and ***M'*** must also be the same. But this is a contradiction, because we assumed that ***M*** and ***M'*** were different m.a.-form marked graphs with the same set of firing sequences. Therefore the m.a.-form is canonic. QED

I will now show how to extend this result to a kind of homomorphic image of a marked graph. This image will turn out to also be a marked graph.

Suppose that we have a marked ***M*** with 20 vertices, say. However, we are interested in only 5 of these vertices. Is it possible to come up with a marked graph ***M'*** of only 5 vertices whose "behavior" mimics the corresponding 5 vertices of ***M*** ? Consider the set of firing sequences S associated with ***M***. If we call the 5 "interesting" vertices the *included* vertices and the other 15 *excluded* vertices, we can define a set of sequences h(S) such that all symbols for excluded vertices have been deleted. Thus, h is a homomorphism from the transition set T to $I \cup \{\lambda\}$ where $I \subseteq T$ is the included set of vertices and λ is the empty string. h is the identity function for included vertices and the "erasure" function for excluded vertices.

The following theorem will show that live marked graphs are closed under this class of "partial-erasing identity homomorphisms".

Theorem II. Given a live marked graph and a designated subset of its vertices, it is possible to reduce the net in such a way that its only vertices are those in the designated subset and the firing sequences are the same as those of the original graph with the symbols for the non-designated vertices removed. Furthermore, this reduced graph is unique by Theorem I.

Proof: Consider an excluded vertex v in the original marked graph with its set of incoming arcs X. $|X|=m$, and its outgoing arcs Y, $|Y|=n$. Advance the simulation of ***M*** to a point where v is enabled. We must remove v from ***M*** in such a way that to the preceding vertices, v looks like it is enabled, and to the succeeding vertices, v looks like it has already fired. This is achieved by the following reduction process:

1) duplicate each input arc (along with its marking) n times, such that $x_i \in X$ becomes x_{ij}, $1 \leq j \leq n$.

2) duplicate each output arc (along with its marking) m times such that $y_i \in Y$ becomes y_{ij}, $1 \leq j \leq m$.

3) eliminate v and splice x_{ij} to y_{ji} calling this new arc z_{ij}. Notice that $M(z_{ij})=M(x_i)+M(y_j)$.

4) reduce the graph to m.a.-form.

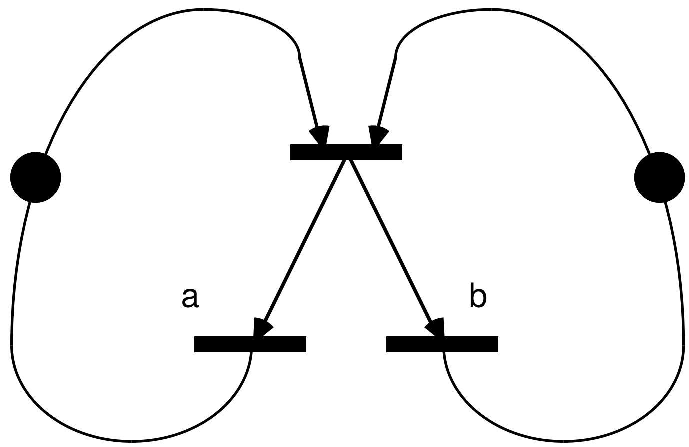
Consider the graph ***M*** and the graph ***M'*** which is ***M*** with v properly cut out. We will call the set of ***M*** 's firing sequences S, and the set of ***M'*** 's firing sequences S'. We first show that $h(S) \subseteq S'$. That is, given a firing sequence $\sigma \in S$, $h(\sigma) \in S'$. Obviously, if we simulate ***M'*** using σ, the simulation will proceed until we reach a "v" symbol in σ. At this point in ***M***, v was enabled and then fired. But this means that each of the successors of v would be enabled by v's firing and the arcs output from v's predecessors would be relieved of some tokens. But we have adjusted things in ***M'*** such that firing v's predecessors enabled v's successors and firing v's successors cleaned up the tokens that firing v would have removed. Therefore $h(S) \subseteq S'$.

We now show that $S' \subseteq h(S)$. This means that given a firing sequence for ***M'***, we can insert "v" in appropriate places to transform it into a firing sequence for ***M***. Assume that σ' is such a firing sequence for ***M***. Start simulating ***M*** with σ'. Either the simulation works or at some point it cannot continue. If the simulation works, we are done. If the simulation stops, it must have been because we were trying to fire a successor w of v and at least one of its input arcs was missing a token. This must be because ***M'*** is exactly the same as ***M*** except those arcs between the predecessors and successors of v. I claim that v is now enabled so that we can fire it and continue with the simulation. Why? Because w was able to fire in ***M'***. But w in ***M'*** had, besides all of its other inputs, *all* of v's inputs. Therefore, if w were able to fire in ***M'***, v must have been able to fire in ***M***. Therefore fire v. w must now be able to fire because all of its inputs from vertices other than v were not affected and w was able to fire in ***M'***. Thus, we can continue with the simulation.

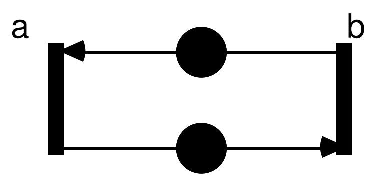
Since $h(S) \subseteq S'$ and $S' \subseteq h(S)$, $h(S)=S'$. QED

Corollary. The image of a safe marked graph may not be safe.

Proof. Reduce



to



QED.

In this paper, then, I have exhibited a canonic form for marked graphs and shown that marked graphs are closed under vertex-deleting homomorphisms.

