A Note on Colored Petri Nets

James L. Peterson

Department of Computer Sciences

University of Texas at Austin

Austin, Texas 78712

TR-136 February, 1980

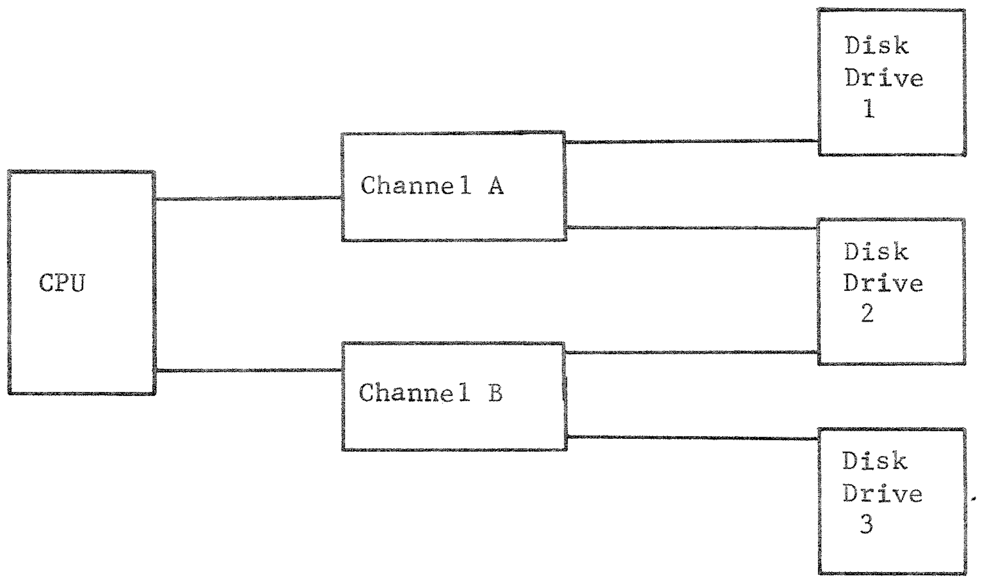
Abstract: We consider the problem of allowing colored tokens in a Petri net. Associating colors with tokens allows concise models of some typical systems. As long as the number of distinct colors is finite, the colored Petri net model is equivalent to the normal Petri net model. Allowing an infinite number of colors extends the model to Turing machine equivalence.

1.0 INTRODUCTION

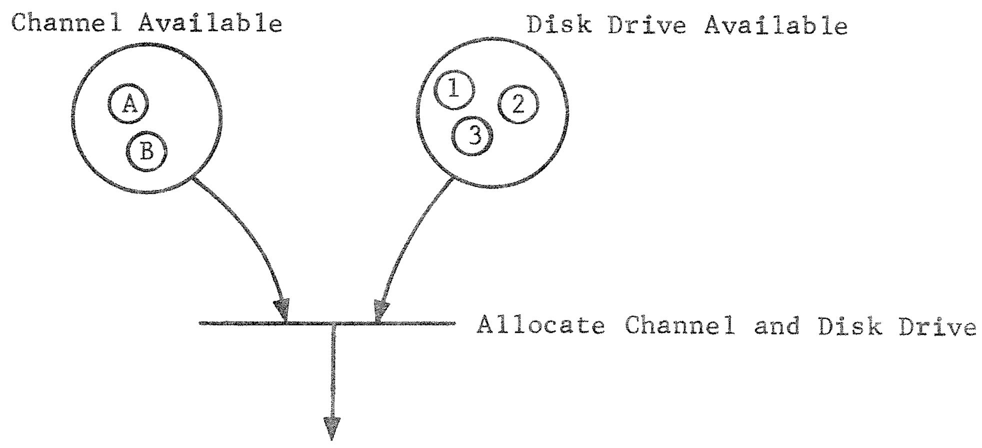
Several extensions to the basic Petri net model have been proposed. Some of these "extensions" (such as self-loops and multiple arcs) do not actually increase the power of the basic model, while other extensions (such as inhibitor arcs or priorities) do increase the class of systems which can be modelled. Recently several researchers have considered extending the model to allow colored or typed tokens [1,2,3,7]. In this note, we consider this extended Petri net model.

We refer the reader to [5] for an introduction to the Petri net model. Briefly, a Petri net is a four-tuple (P,T,I,O) where P is a finite set of places, T is a finite set of transitions, and I and O map T into bags of places. $I(t_j)$ defines the inputs of a transition t_j; $O(t_j)$ defines the outputs of t_j. A marking consists of a distribution of tokens to the places. A transition can fire if each of its inputs has a token (multiple tokens for places which are multiple inputs). The transition fires by removing the enabling tokens from its inputs and adding tokens to all of its output places (multiple tokens for multiple outputs).

When using a Petri net to model a system, tokens often represent objects or resources in the modelled system. As such, these resources may have attributes which are not easily represented by a simple Petri net token. For example, in Figure 1, we show a Petri net which models part of an operating system dealing with a disk system. One place represents the two channels (A and B) while another represents the three disk drives (1, 2 and 3). Although both channels are the same equipment model, and the three disk drives are similarly identical, the



(a) Block Diagram of Shared Disk System



(b) Petri net representation of resources

Figure 1. Using a Petri net to model the resources in a simple computer resource allocation problem.

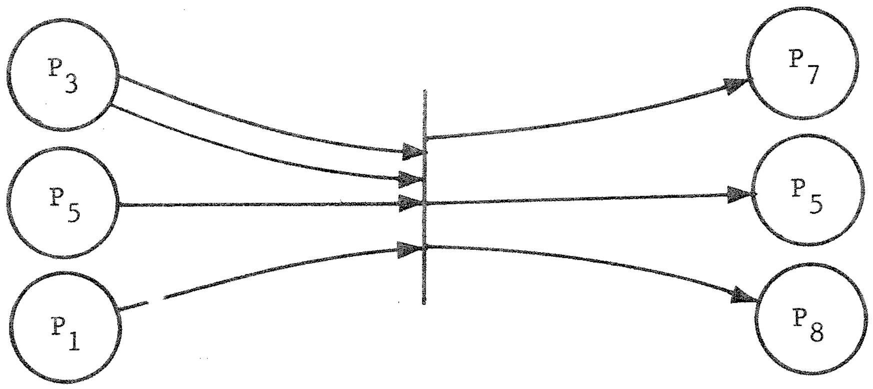
connections between them require that disk drive 1 must use channel A, while drive 3 must use channel B; drive 2 can use either channel A or B.

2.0 COLORED PETRI NETS

To model these systems with a standard Petri net may be difficult, and so we consider extending the model by allowing colored tokens. We define a set of colors, C. A marking now specifies, for each place, the bag of colored tokens at that place. (The use of a bag allows several tokens of the same color to reside in a place.)

The major question now is how to define the firing of a transition: how are the colors of the input tokens used to define the colors of the output tokens? Let q_j be the number of input tokens and r_j be the number of output tokens for a transition t_j. We suggest that the most general firing rule defines for each transition t_j a function f_j of q_j input tokens which produces an r_j-tuple of output tokens. This can be represented by a table such as Figure 2. The q_j left columns represent the input places; the r_j right columns represent the output places. Each row of the table specifies a combination of input colors. If tokens of these colors are available in the corresponding input places, then the transition may fire by removing these enabling tokens and adding a token of the correct color to each output place. A combination of input colors which is not listed does not enable the transition.

This extended Petri net model can easily model many systems which are naturally defined in terms of typed quantities or distinct



Inputs				Outputs		
P_3	P_3	P_5	P_1	P_7	P_5	P_8
red	red	black	blue	yellow	black	blue
red	green	black	blue	orange	black	blue
green	green	black	blue	red	black	blue
.	.	.	.	.	.	.
.	.	.	.	.	.	.
.	.	.	.	.	.	.

Figure 2. An example of the firing rule definition for a colored Petri net.

individuals. Figure 3 is a model of a simple disk scheduling algorithm for the configuration defined in Figure 1. Different individual resources are represented by tokens of differing colors. Notice that transition t_3 is not enabled for undesired input combinations (channel A, drive 3 or channel B, drive 1).

The question is whether this is a significant extension to the Petri net model. We consider two cases: a finite number of colors and an infinite number of colors.

3.0 NETS WITH A FINITE NUMBER OF COLORS

If there are only a finite number, k, of distinct colors, we can convert a net with colored tokens (with the extended firing rule defined above) into an equivalent net with tokens of only one color (with the normal firing rule). Each place in the colored net is mapped into a set of places and each transition is mapped into a set of transitions in the new net. Hence, there is a homomorphic mapping from the new net, its state space and transition sequences into the old net, its state space and transition sequences, and an inverse homomorphic mapping in the other direction.

The new net is created by duplicating the colored net once for each color. Thus, for a place p_i, we define a set of k places $p_{i,1}$, $p_{i,2}$, ..., $p_{i,k}$. We then redefine transitions to interpret a token in $p_{i,c}$ to be a token of color c in place p_i. Specifically, we define for each transition t in the colored net a new transition t′ in the equivalent

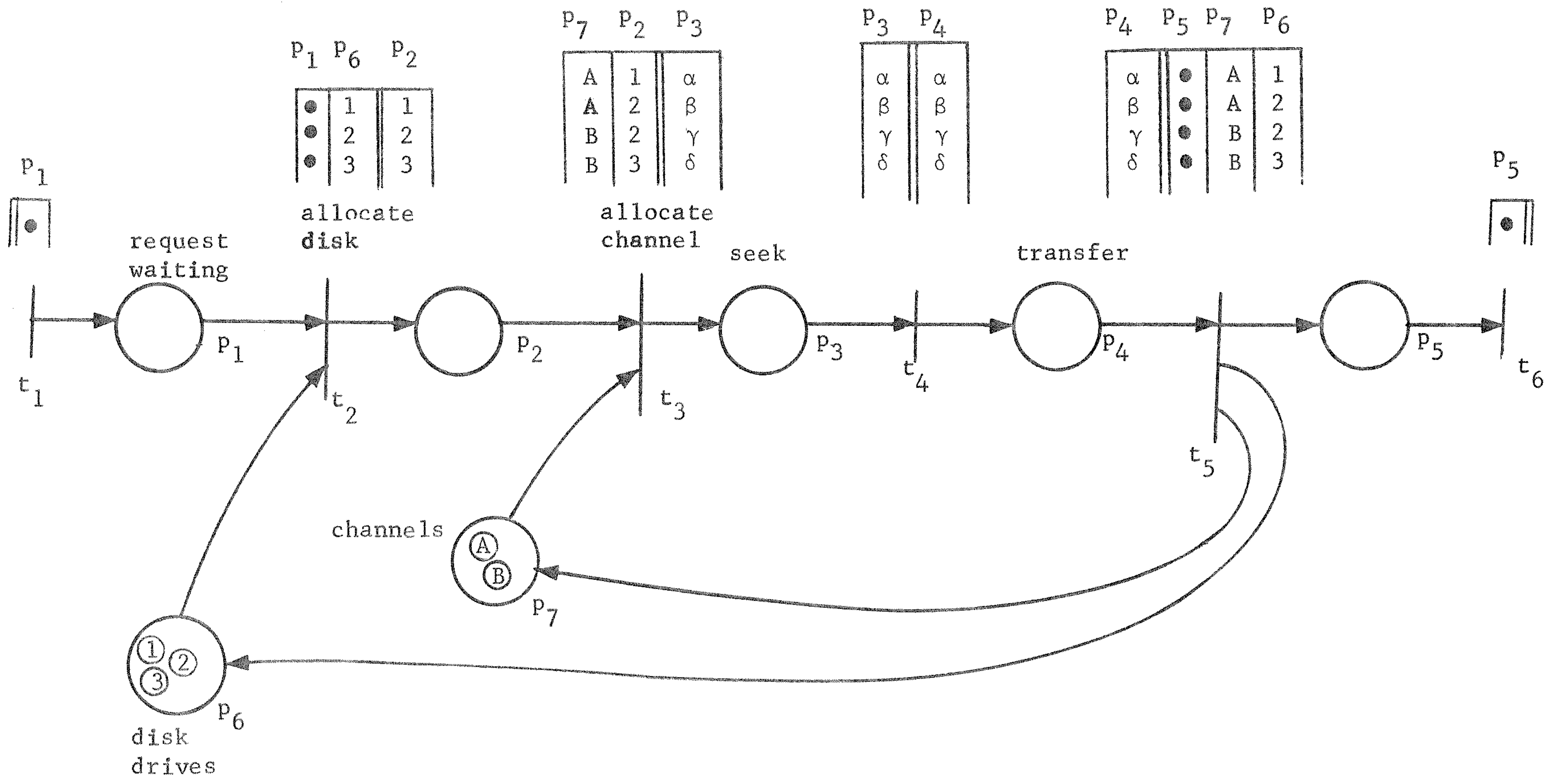


Figure 3. A simple disk scheduling algorithm modeled by a Petri net with colored tokens.

uncolored net. If the row of the table defines the inputs of t to be a token of color c_1 from place p_1, color c_2 from p_2, ..., and color c_q from p_q, then the inputs of t′ are an (uncolored) token from each of p_{1,c_1}, p_{2,c_2}, ..., and p_{q,c_q}. Similarly for outputs of color c_{q+1} to place p_{q+1}, color c_{q+2} to p_{q+2}, ..., color c_{q+r} to p_{q+r}, then we define outputs to $p_{q+1,c_{q+1}}$, $p_{q+2,c_{q+2}}$, ..., and $p_{q+r,c_{q+r}}$.

Figure 4 is the uncolored Petri net equivalent to the colored net of Figure 3.

This construction may produce large numbers of disconnected places which can then be discarded, but in general a colored net with k colors, n places and m transitions will produce an equivalent uncolored net with $n \cdot k$ places and up to $m \cdot k^{q+r}$ transitions, where q is the maximum number of input places and r is the maximum number of output places for any transition.

This construction has also been considered in [3] and [4].

4.0 NETS WITH AN INFINITE NUMBER OF COLORS

In this second case, when the number of colors is allowed to be infinite, it is fairly easy to define a net which uses the colors to represent the positive integers. (Let color c_i represent the integer i, $i > 0$). Now we can use places to simulate registers by representing a value n in a register by a token of color c_n in the place. With our ability to define the action of a transition separately for each color, we can construct transitions to add one to a place, to subtract one from

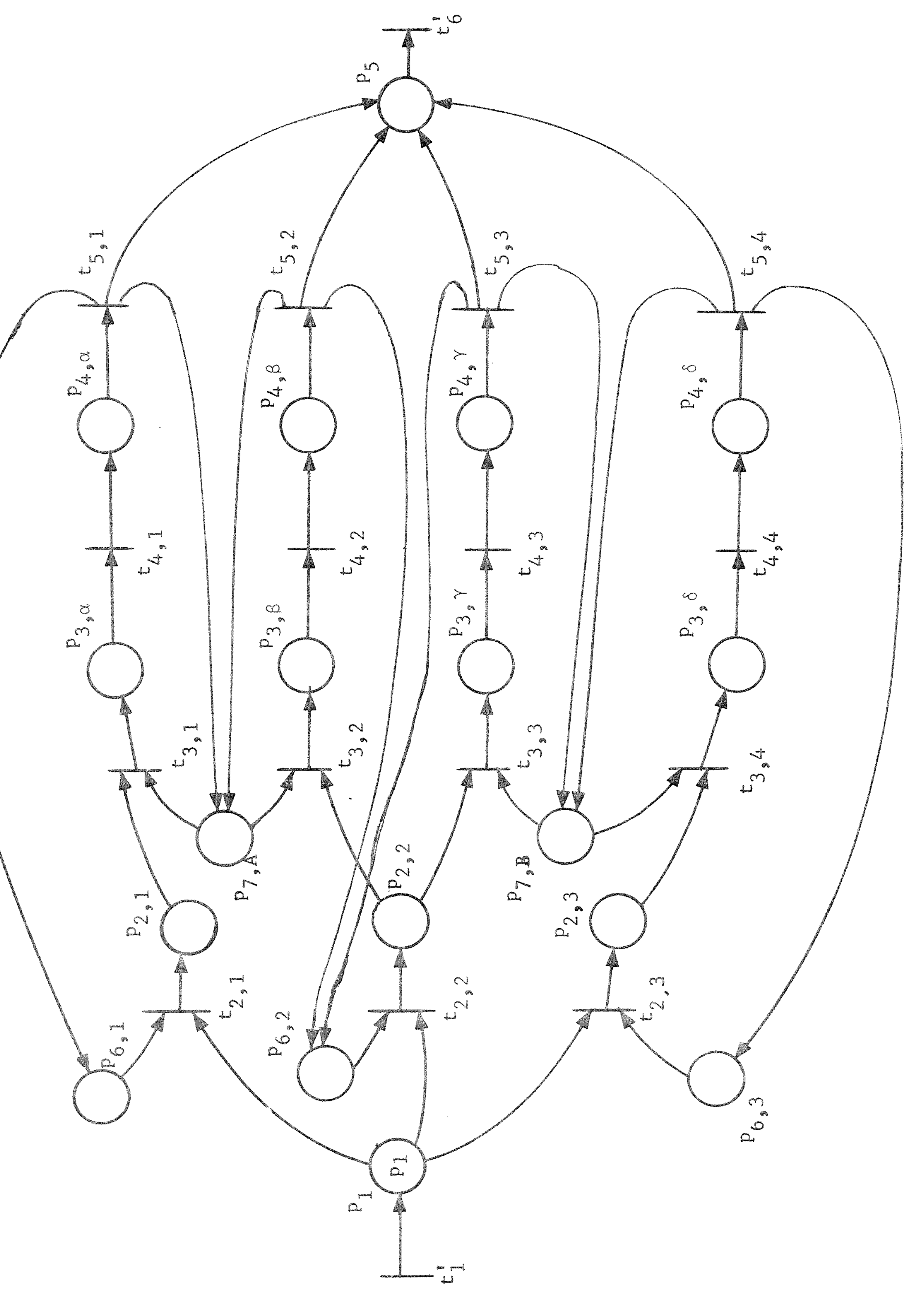


Figure 4. An (uncolored) Petri net which is equivalent to the colored Petri net of Figure 3.

a place, or to test for zero using the color encoding of the integers. These three functions on as few as three registers (places) have been shown to be sufficient for encoding Turing machines. Hence most interesting questions become undecidable. Thus, allowing an infinite number of colors extends the modelling power of the basic Petri net model.

If the transformation given above (for the case of a finite number of colors) is applied to a net with an infinite number of colors, an infinite Petri net is produced. Petri´s thesis [6] showed that infinite nets can model Turing machines.

5.0 SUMMARY

The use of colored tokens in a Petri net model can allow a much more concise model of a system. As long as the number of colors is finite, the model is equivalent to a (much larger) Petri net without colors. However, allowing an infinite number of colors results in an extended model equivalent to a Turing machine, for which most general questions are undecidable.

6.0 REFERENCES

1. F. B. Berlin, "Time-Extended Petri Nets", Master´s Thesis, Department of Computer Sciences, University of Texas, Austin, Texas, (August 1979), 152 pages.

2. H. J. Genrich, K. Lautenbach, and P. S. Thiagarajan, "An Overview of Net Theory", Advanced Course on General Net Theory of Processes and Systems, Fachbereich Informatik, University of Hamburg, Germany, (October 1979).

3. K. Jensen, "Coloured Petri Nets and the Invariant Method", DAIMI PB-104, Computer Science Department, Aarhus University, Denmark, (October 1979), 27 pages.

4. J. L. Peterson, "Modelling of Parallel Systems", Ph.D. Thesis, Department of Electrical Engineering, Stanford University, (December 1973), 254 pages.

5. J. L. Peterson, "Petri Nets", Computing Surveys, Volume 9, Number 3, (September 1977), pages 223-252.

6. C. A. Petri, "Kommunikation mit Automaten", Ph.D. Thesis, University of Bonn, Bonn, Germany, (1962).

7. C. R. Zeros, "Colored Petri Nets: Their Properties and Applications", Technical Report 107, Systems Engineering Laboratory, University of Michigan, Ann Arbor, Michigan, (January 1977), 317 pages.