Bull. Nov. Comp. Center, Comp. Science, 28 (2008), 43–54
© 2008 NCC Publisher

Nets of active resources for distributed systems modeling

V. A. Bashkin*

Abstract. Nets of active resources (AR-nets) are presented. This formalism has the same expressive power as Petri nets but a different syntax: the model is not a bipartite oriented graph, but an oriented graph with two types of arcs (consumption and production). Direction of the arc denotes active and passive participants of the corresponding interaction. The same token may be considered as a passive resource (produced or consumed by agents) and an active agent (producing or consuming resources) at the same time. This model may be useful for systems with dynamic structure of actions.

Several natural modifications of basic syntax are presented, their expressive power is investigated. It is shown that the basic AR-nets and AR-nets with simple firing are equivalent to Petri Nets; AR-nets with simultaneous firing and AR-nets with empty firing ("channels") are Turing-powerful; AR-nets with reset firing are equivalent to Reset Petri Nets.

1. Introduction

Petri nets [8, 7] are one of the most popular formalisms for distributed systems modeling. This is a quite expressive model of parallelism, allowing us to formalize all the basic constructs: sequential composition, nondeterministic choice, parallel composition and synchronization. However, Petri nets are less powerful than Turing machines, so many algorithmic problems are still decidable for them: reachability, safety, liveness etc.

Ordinary Petri nets represent a low-level formalism with a very simple set of basic elements: place, transition, arc and token. It gives no convenient tools for high-level constructs, such as module and hierarchy.

Nowadays there exist a number of Petri net modifications introducing a different high-level syntax, such as Coloured Petri Nets (CPN) [2], Object Nets [10], Nested Petri Nets [6] and many others. Most of them have the same modeling power as ordinary Petri nets (CPN), others are more expressive (Nested Nets).

It is difficult to model with a Petri net a multiagent system with a nonfixed number of agents. The structure of the net is explicitly divided into two classes of elements: places and transitions. The places correspond to the passive component of the system (a *state* or *resources* of the system), the

*This research was partially supported by the Russian Fund for Basic Research (grants 06-01-00106-a and 07-01-00702-a).

transitions correspond to its active component (*actions* or *events* or *agents*). The actual state of the system is defined by a multiset of tokens residing in all places of the net. This multiset can be modified by transition firings. So we can directly model the dynamic change of the passive component of the system (the state transformations). However, it is not possible to model directly the dynamic change of the actions structure, since the set of transitions is fixed and cannot be modified at runtime.

This problem can be easily solved by more sofisticated event modeling: we model an action not by a single transition but by a subnet, containing special service places representing the internal state of an agent (and in particular the number of accessible agents of this type). Obviously, this may be too complex and not convenient, so the task of constructing a more simple syntax for systems with dynamic actions is important.

A natural way to define such a syntax is to use the duality of the Petri net graph. Already in [9] K.-A.Petri noted that

> "In general I would like to say that the exploitation of dualities is a main source of deep insight in any theory of dynamic systems as it is in mathematics; net theory abounds in dualities and this is not by chance."

In [5] K.Lautenbach introduced a notion of dual place/transition nets. In this formalism the transitions are also marked by special tokens called "t-tokens". The meaning of t-tokens is that they prevent transitions from being enabled. A transition carrying a t-token cannot be enabled by any marking of p-tokens. A place in the net can be enabled and fired in a dual way. A place firing transforms the marking of t-tokens, arcs for the place firing are inverted. So the net can be dualized in the obvious way. K.Lautenbach in his work proposed dual P/T nets as a model of a system fault propagation.

In [4] M.Köhler and H.Rölke introduced super-dual nets for modeling with dynamic refinement of events. In this formalism transitions are also marked by special tokens called "pokens", but these pokens *enable* transition firings. Places can also fire, but their firing use a special separate set of arcs called "glow relation" in contrast to common "flow relation". A super-dual net can be dualized by interchanging places and transitions, tokens and pokens, flow arcs and glow arcs. In [4] it is proven that super-dual nets have the same expressive power as ordinary Petri nets.

In both dual P/T nets and Super-Dual nets duality is based on two types of elements of the system — resources and actions (places and transitions). These elements are represented in the net by vertices of a bipartite oriented graph. However, there is another (implicitly) divided set in every Petri net (and in every other bipartite oriented graph) — the set of arcs. It contains arcs of two crucially different types — input arcs from places to transitions remove tokens, output arcs from transitions to places produce tokens. The explicit separation of this notions allows us to define an interesting "orthogonal" syntax for Petri nets.

We dualize the definition of a Petri net. The set of arcs is explicitly transformed into two separate sets of *input arcs* and *output arcs.* The sets of transitions and places are united into a single set of *nodes.* Each node may contain *tokens.* A token in the node may fire, consuming some tokens through input arcs and producing some other tokens through output arcs. So a token simulates behaviour of both an active component (an agent) and a passive component (a resource) at the same time. Therefore the formalism is called "nets of active resources".

The paper is organized as follows. In Section 2 we recall the basic definitions and notations on Petri nets. In Section 3 the basic AR-nets are formally defined and studied. It is shown that AR-nets have the same expressive power as Petri nets. In Section 4 we define several natural extensions of the basic syntax. It is shown that AR-nets with a simple firing are also equivalent to Petri nets; AR-nets with an empty firing (nets with "channels") and AR-nets with a simultaneous firing are Turing-powerful; AR-nets with a reset firing are equivalent to Reset Petri Nets.

2. Preliminaries

Let S be a finite set. A *multiset* m over a set S is a mapping $m : S \to Nat$, where Nat is the set of natural numbers (including zero), i.e. a multiset may contain several copies of the same element.

For two multisets m, m' we write $m \subseteq m'$ iff $\forall s \in S : m(s) \leq m'(s)$ (the inclusion relation). The sum and the union of two multisets m and m' are defined as usual: $\forall s \in S : m + m'(s) = m(s) + m'(s),\ m \cup m'(s) = max(m(s), m'(s))$.

By $\mathcal{M}(S)$ we denote the set of all finite multisets over S.

A *Petri net* is a tuple $N = (P, T, F)$, where

- P is a finite set of *places*;
- T is a finite set of *transitions*, $P \cap T = \emptyset$;
- $F : (P \times T) \cup (T \times P) \to Nat$ is a flow relation (a finite set of *arcs*).

A *marking* in a Petri net is a function $M : P \to Nat$, mapping each place to some natural number (possibly zero). Thus a marking may be considered as a multiset over the set of places.

A *marked Petri net* is a pair (N, M_0), where N is a Petri net and M_0 is its *initial marking.*

Graphically P-elements are represented by circles, T-elements by boxes, and the flow relation F by arrows. Places may carry tokens represented by

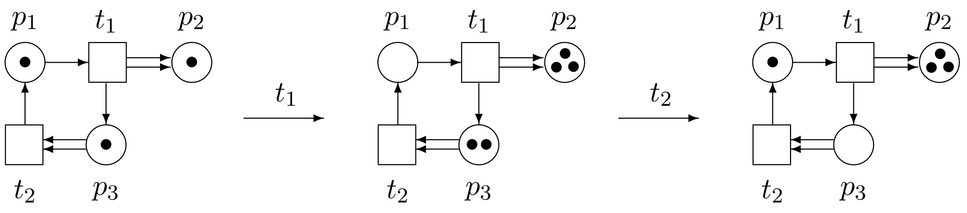

Figure 1. Transition firings in a Petri net

filled circles. A current marking M is designated by putting $M(p)$ tokens into each place $p \in P$.

A transition $t \in T$ is *enabled* in a marking M iff $\forall p \in P\ M(p) \geq F(p,t)$.

An enabled transition t may *fire* yielding a new marking M' s.t. $\forall p \in P$ $M'(p) =_{def} M(p) - F(p,t) + F(t,p)$ (denoted by $M \xrightarrow{t} M'$).

An example of transition firings is given in Figure 1.

A marking M is *reachable* in (N, M_0) iff there exists a finite transition sequence $\sigma \in T^*$ s.t. $\sigma = t_1.t_2 \ldots t_n$ and $M_0 \xrightarrow{t_1} M_1 \xrightarrow{t_2} \cdots \xrightarrow{t_n} M_n = M$. The set of all reachable in (N, M_0) markings (the *reachability set*) is denoted by $R(N, M_0)$.

A place $p \in P$ is *bounded* in (N, M_0) iff $\exists n \in Nat\ \forall M \in R(N, M_0)$ $M(p) \leq n$. A place $p \in P$ is *safe* in (N, M_0) iff $\forall M \in R(N, M_0)\ M(p) \leq 1$.

Petri nets with only bounded places are equivalent to finite automata. Ordinary Petri nets are strictly more powerful than finite automata and strictly less powerful than Turing machines.

An *inhibitor arc* is a special arc from a place to a transition. A transition may be enabled only if there are *no* tokens in its inhibitor precondition. By an inhibitor arc a transition may "observe" the *total* marking of (possibly unbounded) place, so this allows us to model a non-local memory testing. A Turing machine can be simulated by a net with two inhibitor arcs [3].

A *reset arc* is also a special arc from a place to a transition. It doesn't enable or disable a transition. The transition t firing removes *all* tokens from all places linked with t by reset arcs. This is not a non-local memory testing but a non-local memory modification (it is possible to reset a non-fixed number of tokens by a single transition). Reset Petri Nets are strictly more powerful than ordinary Petri nets and strictly less powerful than Turing machines [1] (the boundedness is undecidable but the coverability is decidable).

Pictorially an inhibitor arc is denoted by an arrow with a rounded head (Figure 5), and a reset arc by a crossed arrow (Figure 6).

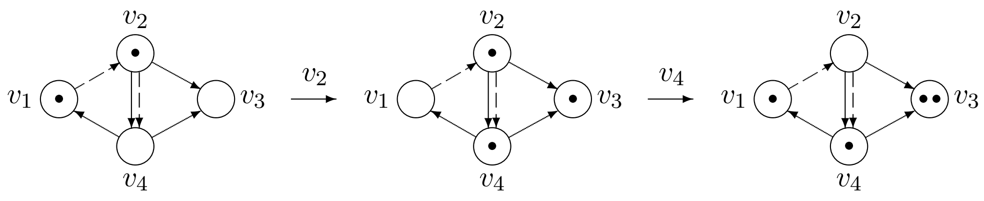

Figure 2. Agent firings in a net of active resources

3. Basic nets of active resources

Definition 1. A *net of active resources* is a tuple $AR = (V, I, O)$, where

- V is a finite set of *resource nodes* (*vertices*);
- $I : V \times V \to \mathit{Nat}$ is a *consumption relation* (*input arcs*);
- $O : V \times V \to \mathit{Nat}$ is a *production relation* (*output arcs*).

In a graphic form, the nodes are represented by circles, the consumption relation by dotted arrows and the production relation by solid arrows (Figure 2).

A *marked net of active resources* is a pair (AR, M_0) where AR is an AR-net and $M_0 : V \to \mathit{Nat}$ is its *initial marking.*

As usual, pictorially the marking is denoted by filled circles.

Definition 2. A resource node $v \in V$ is *active* in a marking M iff

- $M(v) > 0$ (the node v is not empty);
- $\forall w \in V \;\; M(w) \geq I(w, v)$ (there are enough tokens in all its input nodes).

An active node v may *fire* yielding a new marking M' s.t.

$$\forall w \in V \;\; M'(w) =_{def} M(w) - I(w, v) + O(v, w).$$

Some natural notions:

Let $i \in I$ and $i = (v_1, v_2)$. Then the arc i is called an *input* arc for the node v_2 and a *consuming* arc for the node v_1. A token in the node v_1 may be *consumed* through the arc i, a token in the node v_2 can *consume* through the arc i.

Let $o \in O$ and $o = (v_1, v_2)$. Then the arc o is called an *output* arc for the node v_1 and a *producing* arc for the node v_2. A token in the node v_1 can *produce* through the arc o, a token in the node v_2 may be *produced* through the arc o.

It is impossible to define consuming output and producing input. The token may be producing, consuming, produced and consumed at the same

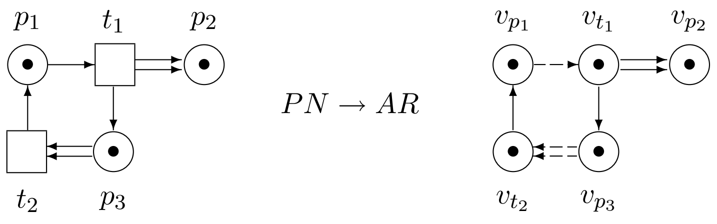

Figure 3. Simulation of a Petri net by an AR-net

time (through different incident arcs). It can even be self-produced or self-consumed.

The syntax of AR-nets differs from the syntax of Petri nets. However, they define the same class of systems:

Theorem 1. *Nets of active resources are equivalent to Petri nets.*[1]

Proof. ($\supseteq$) It is possible to convert a Petri net into an AR-net, converting places and transitions into nodes, arcs from places to transitions into consuming arcs, arcs from transitions to places into producing arcs. The initial marking is extended by putting a single token into every "transition" node. An example of such a transformation is given in Figure 3.

The reachability set of the AR-net is the same as the reachability set of the given Petri net (we do not take into account the marking of a "transition" node, it is a constant).

($\subseteq$) Consider a transformation of an AR-net into a Petri net. We will use a method, proposed in [4] for Super Dual nets.

Each node v of AR-net is converted into a place p_v and a transition t_v, linked by arcs (p_v, t_v) and (t_v, p_v). Each consuming arc (v, w) is converted into the arc (p_v, t_w), each producing arc (v, w) — into the arc (t_v, p_w). Tokens from v are transfered into the place p_v.

An example is given on Figure 4.

4. Extended syntax

In this section we define several natural extensions of the basic AR-nets syntax and investigate their expressive power modulo reachability sets.

4.1. Extended activity

Definition 3. A *simple node* $v \in V$ is active in a marking M iff

- $\forall w \in V \;\; M(w) \geq I(w, v)$.

[1]For each AR-net there exists a Petri net with the same reachability set and vice versa.

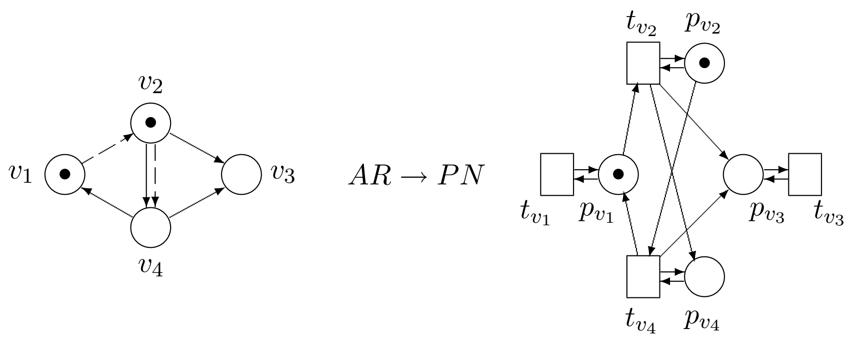

Figure 4. Simulation of an AR-net by a Petri net

The definition of firing for simple nodes is the same as for basic nodes.

Theorem 2.

1. *AR-nets with basic and simple nodes are equivalent to Petri nets.*
2. *AR-nets with only simple nodes are equivalent to Petri nets.*

Proof. ($\supseteq$) A Petri net can be converted into an AR-net with (only) simple nodes by a transformation, similar to the transformation in the first part of the proof of Theorem 1, the only difference is that we may not put the initial token into the simple "transition node".

($\subseteq$) An AR-net with basic and simple nodes can be converted into a Petri net by a modified version of the transformation from AR-nets to Petri nets (the second part of the proof of Theorem 1): for a basic node v we do not add arcs (p_v, t_v) and (t_v, p_v). Obviously, this transformation can be applied both to AR-nets with simple nodes and to AR-nets with basic and simple nodes.

Activity of a simple node does not depend on its marking. A simple node is more similar to a Petri net transition and a Petri net place (at the same time) than to a basic AR-node. Tokens in a simple node are resources, not agents. This syntax extension doesn't actually change the expressive power of the model. So, we can simplify the syntax without loss of expressiveness. However, such a simplification removes the notion of an agent from the model, just like in the case of the classic Petri net syntax.

Definition 4. A *channel* $v \in V$ is active in a marking M iff

- $M(v) = 0$;
- $\forall w \in V \;\; M(w) \geq I(w, v)$.

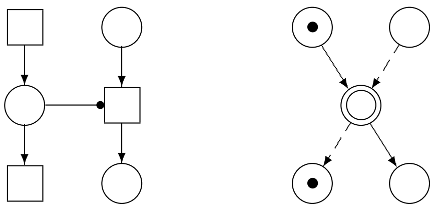

Figure 5. Modeling of an inhibitor arc by a channel

The definition of firing for channels is the same as for basic nodes.

Pictorially we denote channels by double-bordered circles.

Activity of a channel takes into account its complete marking. A channel can "redirect" tokens only being empty itself. This is a non-local memory testing, so the model can be extended up to a Turing machine.

Theorem 3.

1. *AR-nets with basic nodes and at least two channels can simulate Turing machines.*
2. *AR-nets with only channels can simulate Turing machines.*

Proof. Since Petri nets with two inhibitor arcs (linked to different places and transitions) are Turing powerful, it is sufficient to prove that a channel can simulate a separate inhibitor arc.

Consider a transformation of a given Petri net with inhibitor arcs. It is a slightly modified version of the transformation described in the first part of the proof of Theorem 1.

A graphical description of this transformation is given in Figure 5. To illustrate the method, we put to the left a schematic net, containing all possible links of a modeling element to other possible elements of the net. In this specific case we want to define a transformation of a place and a transition, connected by an inhibitor arc. So we also consider the complete set of all possible linked elements. The place is linked with two (types of) transitions — producer and consumer, the transition is linked with two (types of) places — input and output.

The resulting (transformed) net is put to the right. Nodes without labels correspond to the similarly located places and transitions of the original net. A triple "place — inhibitor arc — transition" is replaced by a single channel.

An AR-net with only channels can be constructed by the same scheme of transformation. The only difference is that we replace *all* nodes by channels and remove initial single tokens from "transition nodes".

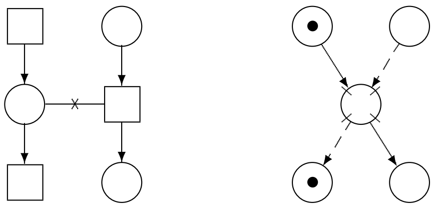

Figure 6. Modeling of a reset arc by a reset node

The immediate corollary of the theorem is the undecidability of all interesting algorithmic problems for nets with channels.

It is easy to see that bounded channels doesn't extend the expressivness of AR-nets since any bounded node can be simulated by a set of basic safe nodes (a separate safe node for every single state of a source node). However, boundedness is undecidable for Turing-powerful Petri net extensions, so we cannot in general check whether a channel is bounded or not.

4.2. Extended firing

Definition 5. A *reset node* $v \in V$ is active in a marking M iff

- $M(v) > 0$;
- $\forall w \in V \;\; M(w) \geq I(w, v)$.

An active reset node v may fire yielding a new marking M' s.t.

$$\forall w \in V \setminus \{v\} \; M'(w) =_{def} M(w) - I(w, v) + O(v, w); \; M'(v) = O(v, v).$$

We denote the reset nodes by crossed circles.

The firing of a single reset token destroys this token itself and all its neighbours. This is a non-local memory modification, so we can extend the expressive power of the model by this syntax modification, but not up to Turing machines (more exactly, to Reset Petri nets [1] with undecidable boundedness and decidable coverability).

Theorem 4.

1. *AR-nets with basic and reset nodes are equivalent to Reset Petri nets.*
2. *AR-nets with reset nodes are equivalent to Reset Petri nets.*

Proof. ($\supseteq$) The scheme in Figure 6 describes the transformation of a reset arc into a reset node.

A net with only reset nodes can be obtained by further replacing of every basic node by a reset node and assigning to every "transition node" v a new additional producing arc (v, v).

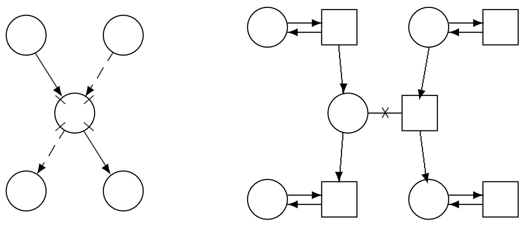

Figure 7. Modeling of a reset node by a reset arc

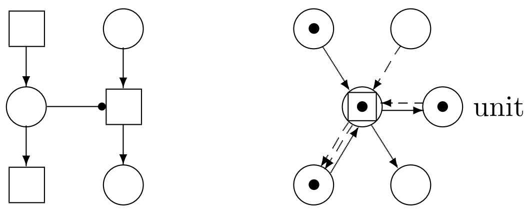

Figure 8. Modeling of an inhibitor arc by a simultaneous node

($\subseteq$) The scheme in Figure 7 describes the backward transformation (a modified version of the first part of the proof of Theorem 1).

Definition 6. A *simultaneous node* $v \in V$ is active in a marking M iff

- $M(v) > 0$;
- $\forall w \in V \quad M(w) \geq M(v) * I(w,v)$.

An active simultaneous node v may fire yielding a new marking M' s.t.

$$\forall w \in V \quad M'(w) =_{def} M(w) - M(v) * I(w,v) + M(v) * O(v,w).$$

We denote simultaneous nodes by squares inside circles.

Simultaneous firing takes into account the total marking of the node. Like a channel, this construct allows us to simulate a Turing machine.

Theorem 5.

1. *AR-nets with basic nodes and at least two simultaneous nodes can simulate Turing machines.*
2. *AR-nets with simultaneous nodes can simulate Turing machines.*

Proof. The scheme in Figure 8 describes the transformation of an inhibitor arc of a Petri net into a simultaneous node.

The simultaneous node always contains one token more than the original inhibitor place. So zero marking of the original place is equivalent to a single token in the simultaneous node. On the other hand, the new resource node "unit" always contains a single token. So the simultaneous node may be active only if it also contains a single token (is "empty").

An AR-net with only simultaneous nodes can be constructed by the same scheme of transformation. The only difference is that we replace all basic nodes by simultaneous ones.

5. Conclusion

We presented a new syntax of Petri nets called nets of active resources. This model is quite simple and compact and allows us to formalize a number of interesting semantic properties, such as dynamic reproduction and dynamic destruction of an agent (including self-reproduction and self-destruction). The basic formalism of AR-nets is just an orthogonal method of a Petri net Representation, so the model can be analyzed by all standard methods of the Petri nets theory.

The AR-models may be useful for distributed systems with non-fixed sets of actors: nets of services, dynamic workflows, business processes. However, this syntax may be less convenient than classical Petri nets in the areas with a strict separation of agents and resources.

We investigated the modeling power of several natural syntax extensions, based on the new concept of active resource. It is shown that the resource-agent representation allows us to increase the expressiveness (even up to Turing machines) without extending the set of node interconnection rules. It is sufficient to use the same fixed set of dependencies — local production and local consumption of a resource.

References

[1] Dufourd C., Finkel A., Schnoebelen Ph. Reset nets between decidability and undecidability // Lect. Notes Comput. Sci. – 1998. – Vol. 1443. – P. 103–115.

[2] Jensen K. Coloured Petri Nets: Basic Concepts, Analysis Methods and Practical Use. – Springer, 1994. – 174 p.

[3] Hack M. Petri net languages. – Computation Structures Group Memo 124. – MIT, 1975.

[4] Köhler M., Rölke H. Properties of Super-Dual Nets // Fundamenta Informaticae. – 2006. – Vol. 72. – P. 245–254.

[5] Lautenbach K. Duality of Marked Place/Transition Nets. – Universitat Koblenz-Landau, 2003.– (Res. Rep. / Institut fur Informatik; N 18).

[6] Lomazova I.A. Nested Petri nets – a Formalism for Specification and Verification of Multi-Agent Distributed Systems // Fundamenta Informaticae. – 2000. – Vol. 43. – P. 195–214.

[7] Peterson J. Petri Net theory and the modeling of systems. – Prentice-Hall, 1981. – 290 p.

[8] Petri C.-A. Kommunikation mit Automaten: PhD thes. – Institute für Instrumentelle Mathematik. – Bonn, 1962.

[9] Petri C.-A. “Forgotten Topics” of Net Theory // Proc. of ATPN’1987. – Lect. Notes Comput. Sci. – Springer-Verlag, 1987. – Vol. 255. – P. 500–514.

[10] Valk R. Petri Nets as Token Objects: An Introduction to Elementary Object Nets // Lect. Notes Comput. Sci. – Springer, 1998. – Vol. 1420. – P. 1–25.