Towards a Notion of Distributed Time for Petri Nets

Extended Abstract

Mogens Nielsen[1], Vladimiro Sassone[2⋆], and Jiří Srba[3⋆⋆]

[1,3] **BRICS**⋆⋆⋆, Dept. of Computer Science, University of Aarhus, DK
{mn,srba}@brics.dk

[2] University of Sussex, UK
vs@susx.ac.uk

Abstract. We set the ground for research on a timed extension of Petri nets where time parameters are associated with tokens and arcs carry constraints that qualify the age of tokens required for enabling. The novelty is that, rather than a single global clock, we use a set of unrelated clocks — possibly one per place — allowing a local timing as well as distributed time synchronisation. We give a formal definition of the model and investigate properties of local versus global timing, including decidability issues and notions of processes of the respective models.

1 Introduction

Verification of finite state systems has been an important area with successful applications to e.g. communication protocols, hardware structures, mobile phones, hi-fi equipment and many others. For systems that operate for example on data from unbounded domains, new methods must be proposed since they are not finite state any more and model/equivalence checking is usually more difficult. Recently algorithmic methods have been developed for process algebras generating infinite state systems [Mol96,BE97], timed process algebra [Yi90], Petri nets [Jan90], lossy vector addition systems [BM99], counter machines [Jan97,AC98], real time systems [ACD90,AD90,AD94,LPY95] and many others. In particular, the idea to equip automata with real time appeared to be very fruitful and there are even automatic verification tools for such systems as UPPAAL [LPY97] and KRONOS [BDM+98].

The main idea behind timed automata is to equip a standard automaton with a number of synchronous clocks, and to allow transitions (a) to be conditioned on clock values, and (b) to affect (reset) clocks. One of the objections to this formalism is the assumption of perfect synchrony between clocks. For

⋆ Author partly supported by MUST project *TOSCA*.
⋆⋆ Author partly supported by the GACR, grant No. 201/00/0400.
⋆⋆⋆ Basic Research in Computer Science, Centre of the Danish National Research Foundation.

J.-M. Colom and M. Koutny (Eds.): ICATPN 2001, LNCS 2075, pp. 23–31, 2001.

many applications this assumption is justified, but for others this is an unrealistic assumption. It is easy to imagine systems which are geographically highly distributed where this is the case, but also within hardware design the issue has been addressed, e.g. within work on so-called Globally Asynchronous Locally Synchronous (GALS) systems [MHKEBLTP98].

We are looking for a formalism in which to model such systems. Petri nets seem to be a natural starting point, since one of the virtues of nets is the explicit representation of locality.

Several models that take time features into account have been presented in the literature (for a survey see [Bow96,Wan98]). For example *timed transitions Petri nets* were proposed by Ramchandani [Ram73]. Here each transition is annotated with its firing duration. Another model where time parameters are associated to the places is called *timed places Petri nets*, introduced by Sifakis [Sif77]. We will analyse *timed-arc Petri nets* [BLT90,Han93], a time extension of Petri nets where time (age) is associated to tokens and transitions are labelled by time intervals, which restrict the age of tokens that can be used to fire the transition. In this model, time is considered to be *global*, i.e., all tokens grow older with the same speed. In spite of the fact that reachability is decidable for ordinary Petri nets [May81], reachability for global timed-arc Petri nets is undecidable [RGdFE99]. On the other hand, coverability is decidable for global timed-arc Petri nets [RdFEA00,AN01]. It is also known that the model offers 'weak' expressiveness, in the sense that it cannot simulate Turing machines [BC89].

We suggest a new model where time elapses in a place independently on other places, taking the view that places represent "localities". We generalise this idea of local clocks in such a way that we allow to define an equivalence relation on places such that two places must synchronise if and only if they are in the same equivalence class. We call this model *distributed timed-arc Petri nets*. As special instances we get *local* timed-arc Petri nets (LT nets) where no places are forced to synchronise, and *global* timed-arc Petri nets (GT nets) with full synchronisation. There is yet another motivation for considering LT nets, namely that they seem to be a weaker model than the original one with global time and some properties could be algorithmically verified. We investigate here to what extent this hope is justified.

2 Distributed Timed-Arc Petri Nets

In this section we define formally the model and we consider both continuous and discrete time.

Definition 1 (Distributed timed-arc Petri net).
A distributed timed-arc Petri net *(DTAPN) is a tuple* $N = (P, T, F, c, E, D)$, *where*

- P *is a finite set of* places,
- T *is a finite set of* transitions *such that* $T \cap P = \emptyset$,

- $F \subseteq (P \times T) \cup (T \times P)$ *is a* flow relation,
- $c : F|_{P \times T} \to D \times (D \cup \{\infty\})$ *is a* time constraint *on transitions such that for each arc* $(p,t) \in F$ *if* $c(p,t) = (t_1, t_2)$ *then* $t_1 \leq t_2$,
- $E \subseteq P \times P$ *is an equivalence relation on places (*synchronisation relation*)*
- $D \in \{\mathbb{R}_0^+, \mathbb{N}\}$ *is either* continuous *or* discrete *time.*

Let $x \in D$ and $c(p,t) = (t_1, t_2)$. We write $x \in c(p,t)$ whenever $t_1 \leq x \leq t_2$. We also define ${}^\bullet t = \{p \mid (p,t) \in F\}$ and $t^\bullet = \{p \mid (t,p) \in F\}$.

Definition 2 (Marking).
Let $N = (P, T, F, c, E, D)$ *be a DTAPN. A* marking M *is a function*

$$M \colon P \to \mathcal{B}(D)$$

where $\mathcal{B}(D)$ *denotes the set of finite multisets on* D.

Each place is thus assigned a certain number of tokens, and each token is annotated with a real (natural) number (*age*). Let $x \in \mathcal{B}(D)$ and $a \in D$. We define $x \mathbin{<\!\!+} a$ in such a way that we add the value a to every element of x, i.e., $x \mathbin{<\!\!+} a = \{b + a \mid b \in x\}$. As *initial markings* we allow only markings with all tokens of age 0.

Definition 3 (Marked DTAPN).
A marked DTAPN *is a pair* (N, M) *where* N *is a distributed timed-arc Petri net and* M *is an initial marking.*

Let us now define the dynamics of DTAPNs. We introduce two types of transition rules: *firing* of a transition and *time-elapsing*.

Definition 4 (Transition rules).
Let $N = (P, T, F, c, E, D)$ be a DTAPN, M a marking and $t \in T$.

- We say that t is *enabled* by M iff $\forall p \in {}^\bullet t.\ \exists x \in M(p).\ x \in c(p,t)$.
- If t is enabled by M then it can be *fired*, producing a marking M' such that:

$$\forall p \in P.\ M'(p) = \Big(M(p) \setminus C^-(p,t)\Big) \cup C^+(t,p)$$

where C^- and C^+ are chosen to satisfy the following equations (note that there may be more possibilities and that all the operations are on multisets):

$$C^-(p,t) = \begin{cases} \{x\} & \text{if } p \in {}^\bullet t \wedge x \in M(p) \wedge x \in c(p,t) \\ \emptyset & \text{otherwise} \end{cases}$$

$$C^+(t,p) = \begin{cases} \{0\} & \text{if } p \in t^\bullet \\ \emptyset & \text{otherwise.} \end{cases}$$

Then we write $M[t\rangle M'$. Note that the new tokens added to places $t^\bullet$ are of the initial age 0.

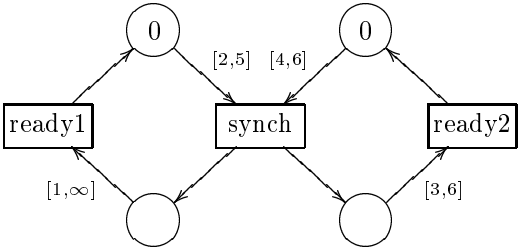

Fig. 1. Example of time synchronisation for GT nets

– We define a *time-elapsing* transition ϵ, for $\epsilon\colon P/E \to D$, as follows, where $[p]_E$ denotes the E-equivalence class of p:

$$M[\epsilon\rangle M' \quad \text{iff} \quad \forall p \in P.\ M'(p) = M(p) \leftarrow \epsilon([p]_E).$$

We write $M \longrightarrow M'$ iff either $M[t\rangle M'$ or $M[\epsilon\rangle M'$ for some t or ϵ.

In particular we can consider the following two classes of DTAPNs. The first one requires an absolute synchronisation and was studied in the past, while the other one is a new model — completely asynchronous.

– *Global timed-arc Petri nets (GT nets)*: $E = P \times P$.
– *Local timed-arc Petri nets (LT nets)*: $E = \Delta_P = \{(p,p) \mid p \in P\}$.

3 Examples

In this section we present three examples of timed-arc Petri nets in order to demonstrate the usefulness of GT nets, LT nets and the general model of distributed timed-arc Petri nets. Let us first consider an example of a GT net. Figure 1 gives its graphical representation.

Places are drawn as circles and squares represent transitions with given names. The flow relation is present in form of arcs and every arc from a place to a transition contains a time interval. In the initial marking a pair of tokens of age 0 is present in the upper two places of the picture. An interesting transition is named 'synch'. This is an example of *time synchronisation*, in the sense that in order to fire this transition from the initial marking, there must be some time-elapsing step by 4 or 5 time units. If the net is considered with continuous time also any ϵ-elapsing step is possible for $4 \leq \epsilon(P) \leq 5$. Then we can fire the transition 'synch'. Whenever we want to fire this transition again, the age of tokens in the places from ${}^\bullet$synch must by synchronised in a similar fashion. Observe that the system can easily deadlock since tokens in places may become *dead*, i.e., they are too old to be useful for firing a transition.

Let us have a look at Figure 2 now. This example is to demonstrate a simple producer/consumer system with continuous time. The net is considered with a local time and whenever a time constraint is missing on an arc, we implicitly

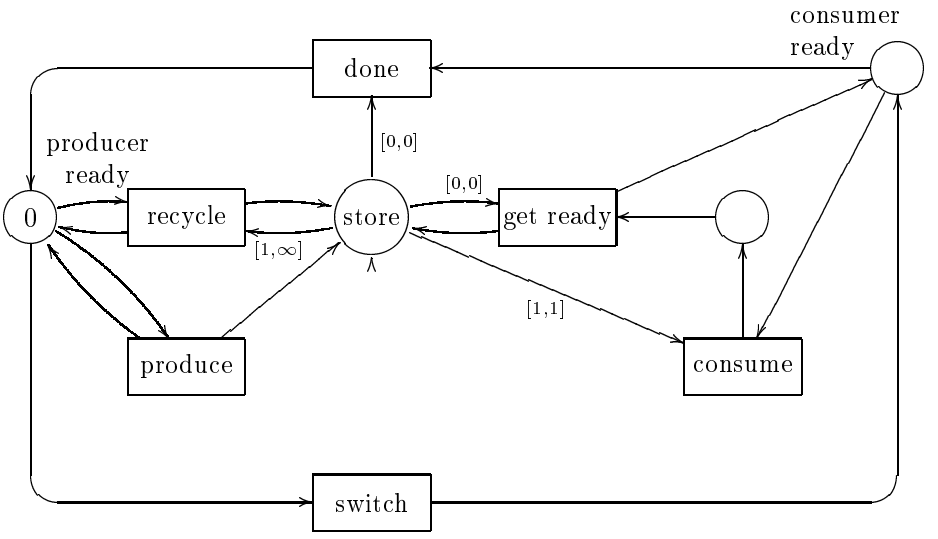

Fig. 2. Producer/consumer example for LT nets

assume that this constraint is irrelevant, i.e., it is of the form $[0, \infty]$. Thus the only interesting place where time-parameters are of importance is 'store'. The other places are just control ones and time can elapse completely independently there. From the initial marking we can fire a transition 'produce' which adds a number of products (tokens) into 'store'. If time elapses during the production process, products of several ages can appear in 'store'. By firing the transition 'switch' we add one token of age 0 into 'store' and one into the place 'consumer ready'. Now producer is active and he can consume products of age 1 from 'store'. Notice that no time elapsing step is allowed, otherwise there is no token of age 0 in 'store' and the transition 'get ready' cannot be fired. When consumer consumed all the products he wanted, a transition 'done' is performed (again checking that there is still the control token of age 0 in 'store') and producer becomes active again. Since consumer is not forced to consume all the products of age 1, it can be the case that products that are too old appear in 'store', however, they can be recycled by firing the transition 'recycle'. The example in Figure 2 demonstrates that LT nets are not so weak as they may look. First, it shows that they allow to implement a potentially *infinite timed-queue* in a place — in our example in the place 'store'. Second, a mechanism is sketched how to *restrict a time-elapsing step by means of a control token* — in our case this token is added by the transition 'switch'.

The last example we will consider is a *Fischer's protocol for mutual exclusion.* Fischer's protocol was suggested by Schneider, Bloom and Marzullo in [SBM92] for testing real-time systems and successfully verified using GT nets by Abdulla and Nylen [AN01]. Figure 3 is taken from the paper [AN01] and it demonstrates a running code for a process i. The idea is that we have potentially infinitely

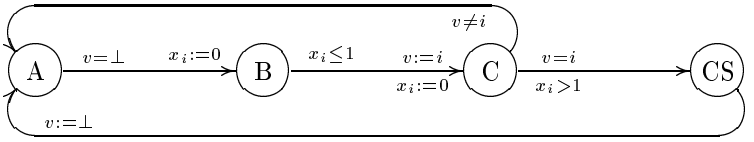

Fig. 3. Fischer's protocol for mutual exclusion

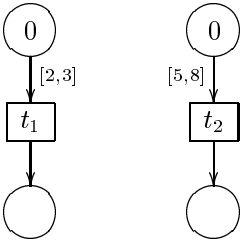

Fig. 4. Dependent transitions in a GT net and independent in an LT net

many processes, each of them running the previously mentioned code. Processes operate on a common shared variable v, A is the initial state and each process has got its own clock x_i. All the clocks are globally synchronised. Our aim is to show that this protocol is correct in the sense that at most one process can enter the critical section CS.

Fischer's protocol can be easily modelled in the GT net formalism as was shown in [AN01]. Synchronisation between places B and C is essential for the mutual exclusion property. However, in this example no behaviour of processes in the critical section is considered. So the protocol only insures safe scheduling mechanism. Assume that we have another GT net N that models the process behaviour in the critical section. If we want e.g. to put the control mechanism together with N and still separate their time-parameters, one solution is to define it as *distributed timed-arc Petri net*. The places in the control mechanism will belong to one equivalence class and the places of N will belong to the other equivalence class. Thus we obtain a complete time-independence between the scheduling process and the process behaviour in the critical section.

4 Investigating DTAPNs

We aim at providing a common ground on which to assess relative expressiveness of GT nets and LT nets. One attempt is to formalise a notion of processes of DTAPNs. The standard notion of processes of P/T nets [GR83] lends itself more readily to LT nets than to GT nets, as illustrated by the net of Figure 4.

Were this net an ordinary, untimed net, we could safely think of the transitions t_1 and t_2 as being completely independent. The situation is not so neat when we consider the time constraints. If we interpret the net as a GT net, i.e., we take the time to be global, after firing t_2, the transition t_1 cannot possibly

fire anymore. So, even if there are no static connections between t_1 and t_2, time constrains do not allow to consider them as totally independent. If instead we consider the net under the local time interpretation, t_1 and t_2 are again independent, as they cannot affect each other's enabledness.

We study DTAPN processes in order to establish their properties with respect to timed firing sequences and to be able to prove results assessing the relative expressiveness of LT versus GT nets.

Another attempt is to consider various decidability questions. Ruiz, Gomez and Escrig recently proved in [RGdFE99] that reachability is undecidable for GT nets. Their proof does not imply undecidability for LT nets, because it relies on synchronised places. In principle, it may seem that the model of LT nets is less powerful than the one of GT nets.

Nevertheless, we demonstrate that reachability for LT nets is undecidable as well. The proof is based on a reduction from the halting problem of Minsky machine with two counters. Notice that this contrasts with the result by Mayr [May81] stating the decidability of reachability for ordinary Petri nets. The reachability problem for local timed-arc Petri nets can be formulated as follows.

Problem: Reachability for LT nets.
Instance: A marked LT net (N, M) and a final marking M'.
Question: $M \longrightarrow^* M'$?

Theorem 1. *Reachability for LT nets is undecidable.*

On the other hand, we only need to restrict the class of considered nets very little in order to get the expected difference between local and global timed nets. Say that a marking is *simple* if each place contains at most one token, and that a marked DTAPN is *simple* if the initial marking is simple.

Theorem 2. *Reachability is decidable for simple LT nets, but undecidable for simple GT nets.*

The coverability problem for GT nets was shown to be decidable — for discrete time in [RdFEA00] and for continuous time in [AN01]. By modifying these results we get that coverability is decidable even for DTAPNs. The problem is defined as follows.

Problem: Coverability for DTAPNs.
Instance: A marked DTAPN (N, M) and a final marking M'.
Question: $\exists M''.\ M \longrightarrow^* M'' \ \wedge\ \forall p \in P.\ M'(p) \subseteq M''(p)$?

Theorem 3. *Coverability for DTAPNs is decidable.*

5 Conclusion

We have introduced a Petri net model aimed at capturing the ideas behind the Globally Asynchronous Locally Synchronous paradigm, and provided some initial results on our model. However, we believe there are many interesting problems to be addressed in the future for such models.

References

[AC98] P. A. Abdulla and K. Cerans. Simulation is decidable for one-counter nets. In *Proceedings of 9th International Conference on Concurrency Theory (CONCUR'98)*, volume 1466 of *LNCS*, pages 253–268, 1998.

[ACD90] R. Alur, C. Courcoubetis, and D. Dill. Model-checking for real-time systems. *5th Symp. on Logic in Computer Science (LICS 90)*, pages 414–425, 1990.

[AD90] R. Alur and D. Dill. Automata for modelling real-time systems. In *Proc. of Int. Colloquium on Algorithms, Languages and Programming*, volume 443 of *LNCS*, pages 322–335, 1990.

[AD94] R. Alur and D. Dill. Automata for Modelling Real-Time Systems. *Theoretical Computer Science*, 126(2):183–236, 1994.

[AN01] P.A. Abdulla and A. Nylen. BQOs and timed Petri nets. In *International Conference on Application and Theory of Petri Nets (ICATPN 2001)*, 2001. `http://www.docs.uu.se/~parosh/publications/publications.shtml`.

[BC89] T. Bolognesi and P. Cremonese. The weakness of some timed models for concurrent systems. Technical Report CNUCE C89-29, CNUCE–C.N.R., 1989.

[BDM+98] M. Bozga, C. Daws, O. Maler, A. Olivero, S. Tripakis, and S. Yovine. Kronos: A model-checking tool for real-time systems. In *Proceedings of CAV'98*, volume 1427 of *LNCS*, pages 546–550. Springer–Verlag, 1998.

[BE97] O. Burkart and J. Esparza. More infinite results. *Bulletin of the European Association for Theoretical Computer Science*, 62:138–159, June 1997. Columns: Concurrency.

[BLT90] T. Bolognesi, F. Lucidi, and S. Trigila. From timed Petri nets to timed LOTOS. In *Proceedings of the IFIP WG 6.1 Tenth International Symposium on Protocol Specification, Testing and Verification (Ottawa 1990)*, pages 1–14. North-Holland, Amsterdam, 1990.

[BM99] A. Bouajjani and R. Mayr. Model checking lossy vector addition systems. In *Annual Symposium on Theoretical Aspects of Computer Science (STACS'99)*, volume 1563 of *LNCS*, pages 323–333. Springer-Verlag, 1999.

[Bow96] Fred D.J. Bowden. Modelling time in Petri nets. In *Proceedings of the Second Australia-Japan Workshop on Stochastic Models*, 1996. `http://www.itr.unisa.edu.au/~fbowden/pprs/stomod96/`.

[GR83] U. Goltz and W. Reisig. The non-sequential behaviour of Petri nets. *Information and Computation*, 57:125–147, 1983.

[Han93] H.M. Hanisch. Analysis of place/transition nets with timed-arcs and its application to batch process control. In *Application and Theory of Petri Nets*, volume 691 of *LNCS*, pages 282–299, 1993.

[Jan90] P. Jancar. Decidability of a temporal logic problem for Petri nets. *Theoretical Computer Science*, 74(1):71–93, 1990.

[Jan97] P. Jancar. Bisimulation equivalence is decidable for one-counter processes. In *Automata, Languages and Programming, 24th International Colloquium (ICALP'97)*, volume 1256 of *LNCS*, pages 549–559. Springer-Verlag, 1997.

[LPY95] K.G. Larsen, P. Pettersson, and W. Yi. Model-checking for real-time systems. In *Proc. of Fundamentals of Computation Theory*, number 965 in LNCS, pages 62–88, 1995.

[LPY97] K.G. Larsen, P. Pettersson, and W. Yi. UPPAAL in a Nutshell. *Int. Journal on Software Tools for Technology Transfer*, 1(1–2):134–152, 1997.

[May81] E.W. Mayr. An algorithm for the general Petri net reachability problem (preliminary version). In *Proc. 13th Ann. ACM Symposium on Theory of Computing*, pages 238–246. Assoc. for Computing Machinery, 1981.

[MHKEBLTP98] T. Meincke, A. Hemani, S. Kumar, P. Ellervee, J. Berg, D. Lindqvist, H. Tenhunen, A. Postula. Evaluating Benefits of Globally Asynchronous Locally Synchronous VLSI Architecture). In *Proc. 16th Norchip*, pages 50–57, 1998.

[Mol96] F. Moller. Infinite results. In *Proceedings of CONCUR'96*, volume 1119 of *LNCS*, pages 195–216. Springer-Verlag, 1996.

[Ram73] C. Ramchandani. *Performance Evaluation of Asynchronous Concurrent Systems by Timed Petri Nets*. PhD thesis, Massachusetts Institute of Technology, Cambridge, 1973.

[RdFEA00] V. Valero Ruiz, D. de Frutos Escrig, and O. Marroquin Alosno. Decidability of properties of timed-arc Petri nets. In *ICATPN 2000*, volume 1825, pages 187–206, 2000.

[RGdFE99] V. Valero Ruiz, F. Cuartero Gomez, and D. de Frutos Escrig. On non-decidability of reachability for timed-arc Petri nets. In *Proceedings of the 8th Int. Workshop on Petri Net and Performance Models (PNPM'99)*, pages 188–196, 1999.

[SBM92] F. B. Schneider, B. Bloom, and K. Marzullo. Putting time into proof outlines. In *Proceedings REX Workshop on Real-Time: Theory in Practice*, volume 600 of *LNCS*, pages 618–639. Springer-Verlag, 1992.

[Sif77] J. Sifakis. Use of Petri nets for performance evaluation. In *Proceedings of the Third International Symposium IFIP W.G. 7.3., Measuring, modelling and evaluating computer systems (Bonn-Bad Godesberg, 1977)*, pages 75–93. Elsevier Science Publishers, Amsterdam, 1977.

[Wan98] J. Wang. *Timed Petri Nets, Theory and Application*. Kluwer Academic Publishers, 1998.

[Yi90] Wang Yi. Real–time behaviour of asynchronous agents. In *Proceedings of the International Conference on Concurrency Theory (CONCUR'90)*, number 458 in LNCS. Springer–Verlag, 1990.