Small Polynomial Time Universal Petri Nets

Dmitry A. Zaitsev
International Humanitarian University
Department of Computer Engineering
Fontanskaya Doroga, 33, Odessa 65009, Ukraine
daze@acm.org

The time complexity of the presented in 2013 by the author small universal Petri nets with the pairs of places/transitions numbers (14,42) and (14,29) was estimated as exponential. In the present paper, it is shown, that their slight modification and interpretation as timed Petri nets with multichannel transitions, introduced by the author in 1991, allows obtaining polynomial time complexity. The modification concerns using only inhibitor arcs to control transitions’ firing in multiple instances and employing an inverse control flow represented by moving zero. Thus, small universal Petri nets are efficient that justifies their application as models of high performance computations.

1 Introduction

The first explicitly constructed universal Petri net (UPN) [9] consists of thousands of vertices and works exponentially slow; for its design, a direct simulation of Petri net behavior according to its state equation was applied. The construction was done within the class of pure inhibitor Petri nets.

Small universal Petri nets with the pairs of places/transitions numbers (14,42) [12] and (14,29) [11] were constructed using the technique of simulating known small universal Turing machines (UTM): the net UPN(14,42) simulates UTM(6,4) [3, 4] and the net UPN(14,29) simulates weakly universal Turing machine (2,4) – WUTM(2,4) [3, 5]. The construction was done within the class of deterministic inhibitor Petri nets. Their time complexity was estimated as exponential.

In the present work, these nets are modified: a place LEFT added to represent the left moves of the TM control head in an explicit way; only inhibitor arcs are used to control transitions which fire in multiple instances; the control flow is represented in an inverse way with moving zero to provide check via inhibitor arcs. Interpretation of the modified UPN as a Petri net with multichannel transitions [7, 8] gives polynomial estimations of time complexity. Thus, constructed UPN are efficient models of computations.

In sequel, net PolyUPN(15,29) is constructed on UPN(14,29), and it is shown that interpreting its behavior as a deterministic timed inhibitor Petri net which transitions’ firing time is discrete and equal to unit and unlimited number of transitions’ channels gives polynomial time complexity; the corresponding class of Petri net is named a deterministic arithmetic Petri net. In the same way the results could be obtained for other early constructed universal Petri nets.

2 A deterministic arithmetic Petri net

A class of deterministic arithmetic Petri net is employed in the present paper that represents a combination of a deterministic inhibitor Petri net (DIPN) [12, 11] and a timed Petri net with multichannel transitions [7, 8] for the case when the firing time of each transition is equal to unit. Since all the times
are equal to unit, it is more convenient to not introduce the time concept at all but to perceive Petri net as a synchronous [6] net when at each step the maximal valid set of firable transitions fires. The validity of the maximal set is defined as not leading to negative markings. But when using transitions’ numbers as priorities in deterministic inhibitor Petri net, only one transition could fire at a step possibly in a few instances. The necessity of restricting the net behavior making it deterministic is stipulated by the deterministic character of simulated TM.

A deterministic arithmetic Petri net (DAPN) is in bipartite directed multigraph supplied with a dynamic process that has synchronous deterministic behavior. DAPN is denoted as a quadruple \( N = (P,T,F,\mu^0) \), where \( P \) and \( T \) are disjoint sets of vertices called places and transitions respectively, the mapping \( F \) defines arcs between vertices, their type and multiplicity, and the mapping \( \mu^0 \) represents the initial state (marking). The transition choice order is defined by their enumeration \( T = \{t_1,t_2,\ldots,t_n\} \), \( n = |T| \); places are also supposed been enumerated \( P = \{p_1,p_2,\ldots,p_m\} \), \( m = |P| \), and the places’ marking is represented as a vector \( \mu, \mu_j = \mu(p_j) \) with integer index \( j, 1 \leq j \leq m \), where \( \mu_j \) is a nonnegative integer equal to the number of tokens situated in place \( p_j \).

The mapping \( F : (P \times T) \rightarrow \{0,w^-,1\} \cup (T \times P) \rightarrow \{0,w^+\} \) defines arcs, their types and multiplicities, where transition input and output arc multiplicities \( w^- \) and \( w^+ \) are natural numbers, a zero value corresponds to the arc absence, a positive value – to the regular arc with indicated multiplicity, and a minus unit – to the inhibitor arc. As it was shown in [6], Petri nets with the multiple arcs are easily converted to ordinary Petri nets with regular arcs’ multiplicity equal to unit. To avoid nested indices we denote \( w^- (p_j,t_i) \) as \( w^-_{j,i} \) and \( w^+ (t_i,p_k) \) as \( w^+_{i,k} \).

In graphical form, places are drawn as circles and transitions as rectangles. An inhibitor arc is represented by a small hollow circle at its end, and a small solid circle (read arc) represents the abbreviation of a loop. Regular arc’s multiplicity greater than unit is inscribed on it and place’s marking greater than zero is written inside it.

To estimate firability conditions on each incoming arc of a transition, the following auxiliary operation is defined

\[
x \div y = \begin{cases} \quad x \div y & \text{if } y > 0, \\ 0 & \text{if } y = -1, x > 0, \\ \infty & \text{if } y = -1, x = 0. \end{cases}
\]

To avoid inconsistency with infinite number of instances, here we prohibit transitions without input regular arcs.

The behavior (dynamics) of a DAPN could be described by the corresponding state equation similarly to [9,7,8]. The present work considers the behavior as result of sequential applying the following transition firing rule:

1. net transitions \( t_i \) are checked sequentially with integer index \( i \) ranging from 1 to \( n \);
2. the number of instances of transition \( t_i \) firable at the current step is equal to

\[
\nu_i = \nu(t_i) = \min_{j}(\mu_j \div w^-_{j,i}), \quad 1 \leq j \leq m, \quad F(p_j,t_i) \neq 0.
\]
3. the first transition \( t_i \) (with the minimal value of index \( i \)) having \( \nu_i > 0 \) fires in \( \nu_i \) instances;
4. when transition \( t_i \) fires, it
   
   (a) extracts \( \nu_i \div w^-_{j,i} \) tokens from each its input place \( p_j \) for regular arcs \( F(p_j,t_i) > 0 \);
   
   (b) puts \( \nu_i \div w^+_{i,k} \) tokens into each its output place \( p_k \), \( F(t_i,p_k) > 0 \);
5. the net halts if firable transitions are absent.
Table 1: WUTM(2,4) behavior and its encoding

| $\Sigma \setminus \Omega$ | $u_1$ | $u_2$ |
|--------------------------|------|------|
| $s(\Sigma) \setminus s(\Omega)$ | 0    | 1    |
| 0                        | 3, left, 0 | 4, right, 0 |
| 1                        | 4, left, 1 | 3, left, 1 |
| $\emptyset$              | 4, left, 0 | 1, right, 1 |
| 1                        | 4, left, 0 | 2, right, 1 |

When a transition, having a single regular incoming arc with multiplicity $x$ from place $p$ and a single regular outgoing arc with multiplicity $y$ to place $p'$, fires, it implements the following computations $\mu(p) := \mu(p) \mod x$, $\mu(p') := \mu(p') + y \cdot (\mu(p) \div x)$. That is why it is named arithmetic Petri net: it implements division by $x$ and multiplication by $y$. Choosing either $x$ or $y$ equal to unit we obtain either pure multiplication or pure division.

Thus, transitions could be thought of as virtual actions. The number of really started actions depends on the amount of available resources represented by transitions’ input places. Why we should restrict the number of transitions’ instances to unit and fire them in sequence, as in classical Petri net, when available resources allow firing them simultaneously? Anyway, classical sequential order of transitions’ firing could be obtained as a special case attaching a place to each transition connected with read arc and having marking equal to unit.

3 Constructing PolyUPN(15,29)

As in UPN(14,29) \cite{12}, the source information for simulation is the transition function of WUTM(2,4) \cite{3, 5} and the encoding of its states and tape symbols given by Table 1. In WUTM(2,4), an infinite repetition of definite blank words is written on its tape: $w_l = 00\ldots0$ to the left and $w_r = 0\ldots0\ldots0\ldots0\ldots01$ to the right of the working zone. According to the function $s(x_l \ldots x_2 x_1) = \sum_{i=0}^{l-1} s(x_i) \cdot 5^i$ for the tape words’ encoding \cite{12}, the codes of the left and right blank words are: $sw_l = 167$ and $sw_r = 13596$.

To obtain PolyUPN(15,29), net UPN(14,29) \cite{12} was modified in the following way:

- place LEFT ($p_{15}$) added, to represent the left moves of the control head, together with arcs from transitions simulating instructions with the left move: $t_4, t_6, t_7, t_8, t_{10}$; thus a pair of places LEFT ($p_{15}$) and RIGHT ($p_7$) are complimentary: only one of them contains a token when simulating a TM step;
- a regular arc is added from place LEFT ($p_{15}$) to transition $t_{29}$ to clean it after simulating the left move;
- the control flow within the sequence of subnets $MASL, MDSL$ represented by the sequence of vertices $p_6, t_{13}, p_{10}, t_{16}, p_{11}, t_{19}, p_8, t_{22}, p_{12}, t_{25}, p_{13}, t_{28}$ or $t_{29}$ was modified; a token was put into each place; reverse order of arcs was used; inhibitor arcs from the previous places were added; thus, the control flow is represented by moving zero marking;
- all the checks of control flow places on unit with read arcs are replaced by checks on zero with inhibitor arcs;
- all the checks of place RIGHT ($p_7$) on unit with read arcs are replaced by checks of place LEFT ($p_{15}$) on zero;
all the arcs from transitions, simulating TM instructions $t_3 - t_{10}$ to place $MOVE$, were reversed in their direction.

The general scheme of PolyUPN(15,29) is shown in Fig. 1. Subnets are depicted as rectangles with double line border. Some vertices have mnemonic names besides their numbers. Used subnets $FS$, $MA5LR$, and $MD5LR$ are represented in Fig. 2, Fig. 3, Fig. 4 correspondingly. Places with the same name (number) are considered as the same place and should be merged all over the components. The obtained final assembly of PolyUPN(15,29) is shown in Fig. 5. Since it is rather tangled, it could be represented in a tabular form similar to UPN(14,29) [12].

Place $U$ contains encoded TM state $s(u)$, place $X$ contains encoded current cell symbol $s(x)$, and places $L$ and $R$ contain encoded left and right parts of the tape working zone respectively regarding the current cell. At the beginning of each computation step, place $STEP$ launches subnet $FS$, which simulates WUTM(2,4) transition function. Subnet $FS$ produces the encoding $s(u')$ of the new state and the encoding $s(x')$ of the new symbol in places $U$ and $X$ respectively to simulate the Turing machine instruction. Subnet $FS$ also puts a token into either place $RIGHT$ or place $LEFT$ to indicate the control head right or left moves correspondingly. A token is extracted from place $MOVE$ after subnet $FS$ has finished that launches the sequence of subnets $MA5LR, MD5LR$, which simulates the control head moves. At the end of a simulated computation step, a token is put into place $STEP$ that allows the simulation of the next instruction to begin. Moreover, places $LEFT$ and $RIGHT$ are cleared.

Subnet $FS$ (shown in Fig. 2) simulates the transition function of WUTM(2,4) as follows: the Turing machine instruction for each pair $(x,u)$ is encoded by a transition with the label $(s(x), s(u))$ in the subnet $FS$; its input arcs from places $X$ and $U$ have corresponding multiplicity, zero multiplicity means the arc absence. For a TM instruction $(x,u,x',v,u')$, output arcs to places $X$ and $U$ have multiplicity $s(x')$ and $s(u')$ correspondingly with an extra arc either to place $RIGHT$ when $v = right$ or place $LEFT$ when $v = left$. Subnet $FS$ of PolyUPN(15,29) is represented in Fig. 2, each transition has an incoming arc from place $STEP$ and place $MOVE$. The required relations of priorities are shown via dashed auxiliary arcs connecting transitions, providing only one firable transition on each step of UPN work.

The TM tape is represented by the places $L$, $X$, and $R$ containing the encoded left part of the working zone, the current cell symbol and the encoded right part of the working zone correspondingly. The moves of the tape head on the tape are simulated by the two connected subnets $MA5LR$ and $MD5LR$ shown in Fig. 3, Fig. 4. The meaning of subnets’ names is the following: $MA5$ multiplication and addition with radix 5 ($S := S \cdot 5 + X$), $MD5$ modulo and division with radix 5 ($S := S \text{div} 5, X := S \mod 5$); $LR$ choice of places either $L$ or $R$, where codes of the left and right parts of the tape, regarding the current cell symbol code $X$, are stored, depending on the marking of places $LEFT$ and $RIGHT$. Two transitions $lb$
and \( rb \) simulate peculiarities of weakly universal TM work, they add the blank word codes \( sw_L \) and \( sw_R \), to the codes \( L \) and \( R \) of the left and right parts of the tape working zone correspondingly when its value is equal to zero.

Thus, the sequence of subnets \( MA5LR, MD5LR \) implements the following operations:

- to simulate a left move (when place \( RIGHT = 0 \)): \( R := R \cdot 5 + X, L := L \div 5, X := L \mod 5 \);
- to simulate a right move (when place \( LEFT = 0 \)): \( L := L \cdot 5 + X, R := R \div 5, X := R \mod 5 \).

PolyUPN(15,29) is composed according to Fig. 1 via inserting subnets and merging places with the same names. Transitions are enumerated to provide required relations of priorities; places are enumerated in an arbitrary order; the number of arcs is 125. The obtained PolyUPN(15,29) is shown in Fig. 5.

The work of PolyUPN(15,29) was simulated in the environment of system Tina (http://laas.fr/tina) in the step-by-step order; the results coincide with the tracing table of [12]. Tina does not provide firing transitions in a few instances but the conditions for such firing were preserved: the only firable transition has been firing sequentially during amount of steps equal to the number of its instances in DAPN.

4 Estimations of PolyUPN(15,29) time complexity

Lemma 1. Subnet \( FS \) simulates TM transition function in a single DAPN step.

The proof of the lemma is an immediate conclusion of subnet \( FS \) arrangement rules [12] [11] and late modifications regarding adding place \( LEFT \), reversing arcs of place \( MOVE \), and initial marking of place \( MOVE \) equal to unit. When \( STEP = 0 \), none of \( FS \) transitions is firable; when \( STEP = 1 \), only one transition is firable that fires removing a token from places \( STEP \) and \( MOVE \) and changing marking of places \( U \) and \( X \) according to TM transition function.
Figure 3: Basic subnet of the tape encoding $MA5LR$ – add a symbol to the code

Figure 4: Basic subnet of the tape decoding $MD5LR$ – extract a symbol from the code
Figure 5: UPN(14,29) in graphical form
Lemma 2. The sequence of subnets $\text{MASLR}$, $\text{MD5LR}$, supplied with transitions $\text{lb}$, $\text{rb}$, simulates work
with weakly universal TM tape in not more that 13 DAPN steps.

Proof. Before simulating current TM step, only one of places $L$, $R$ can become zero: $L$ – for the left
move and $R$ – for the right move as a result of division operation that means hitting the corresponding
edge of the tape working zone. Then place $\text{STEP}$ enables one of transition $\text{lb}$, $\text{rb}$ which fires before $\text{FS}$
work and inserts the corresponding blank word code disabling the fired transition. Then, as $\text{lb}$, $\text{rb}$ are
disabled, subnet $\text{FS}$ starts.

After $\text{FS}$ finished, marking of place $\text{MOVE}$ is zero, marking of places $U$ and $X$ corresponds to the
new state/symbol pair and place $\text{LEFT}$ is marked for instructions with the left move while place $\text{RIGHT}$
marked for instructions with the right move. Let us consider an instruction with the left move when
$\text{LEFT} = 1, \text{RIGHT} = 0$.

Subnet $\text{MASLR}$ works first. The only firable transition $t_{11}$ fires in $R$ instances putting $R \cdot 5$ tokens into
place $p_9$. Then $t_{13}$ fires taking a token from $p_{10}$ and putting a token to $p_6 (\text{MOVE})$ that looks like moving
zero from $p_6 (\text{MOVE})$ to $p_{10}$. The only firable transition $t_{14}$ fires in $R \cdot 5$ instances putting $R \cdot 5$ tokens
into place $R$ ($R := R \cdot 5$). Then $t_{16}$ fires moving zero from $p_{10}$ to $p_{11}$. The only firable transition $t_{17}$ fires
in $X$ instances adding $X$ tokens into place $R$ and cleaning place $X$. Then $t_{19}$ fires moving zero from $p_{11}$ to
$p_8 (\text{MOVE1})$ that disables transitions of subnet $\text{MASLR}$ and enables transitions of subnet $\text{MD5LR}$. The result
of subnet $\text{MASLR}$ work is the following $X := 0, R := R \cdot 5 + X$ and marking of internal place $p_8$ is
equal to zero that is achieved via the only transitions firing sequence $t_{11} t_{13} t_{14} t_{16} t_{17} t_{19}$. As the control flow
places $p_6, p_{10}$, $p_{11}$ contain a token, none of $\text{MASLR}$ transitions is enabled.

Then subnet $\text{MD5LR}$ works. The only firable transition $t_{21}$ fires in $L \text{ div } 5$ instances putting $L \text{ div } 5$
tokens into place $p_{14}$. Then $t_{22}$ fires moving zero from $p_8 (\text{MOVE1})$ to $p_{12}$. The only firable transition
$t_{24}$ fires in $L \text{ mod } 5$ instances putting $L \text{ mod } 5$ tokens into place $X$ ($X := L \text{ mod } 5$). Then $t_{25}$ fires
moving zero from $p_{12}$ to $p_{13}$. The only firable transition $t_{27}$ fires in $L \text{ div } 5$ instances returning $L \text{ div } 5$
tokens into place $L$. Then $t_{28}$ fires taking a token from place $\text{LEFT}$ and putting a token into each
place $p_{13}$ and $p_5 (\text{STEP})$ that disables transitions of subnet $\text{MD5LR}$ and enables transitions of subnet
$\text{FS}$. The result of subnet $\text{MASLR}$ work is the following $X := L \text{ mod } 5, L := L \text{ div } 5, \text{LEFT} := 0$ and
marking of internal place $p_{14}$ is equal to zero that is achieved via the only transitions firing sequence
$t_{21} t_{22} t_{24} t_{25} t_{27} t_{28}$. As the control flow places $p_8, p_{12}, p_{13}$ contain a token, none of $\text{MASLR}$
transitions is enabled.

In the same way the proof could be composed for an instruction with the right move when $\text{RIGHT} = 1,
\text{LEFT} = 0$. The only transitions firing sequence is $t_{11} t_{13} t_{14} t_{16} t_{17} t_{19} t_{21} t_{22} t_{24} t_{25} t_{27} t_{28}$. In the
both cases the number of transitions equals to 12; the number of their firing insrances, when it is greater
than unit, is written as a superscript. Thus, not more than 13 transitions fire.

Theorem 1. UPN(14,29) simulates WUTM(2,4) in time $O(14 \cdot k)$ and space $O(k)$, where $k$ is the number
of WUTM(2,4) steps.

Theorem 1 is an immediate conclusion of lemma 1, lemma 2 and analogous theorem proven in 12
for UPN(14,29).

5 Conclusions

A deterministic arithmetic Petri net PolyUPN(15,29) with 15 places and 29 transitions was constructed,
that simulates weakly universal Turing machine of Neary and Woods 3, 5 with 2 states and 4 symbols.
in linear time. Consequently it simulates a given TM in polynomial time.

Moreover, considering the chain of translations described in [12], we conclude that PolyUPN(15,29) simulates a given DIPN in polynomial time. Strict complexity estimations require encoding of a given DAPN with gliders of cellular automaton 110 [1,2] because WUTM(2,4) simulates its work. In this concern, DAPN could be translated in polynomial time to either of: gliders, cyclic tag system, 2-tag system, Turing machine, or bi-tag system.

Thus, small universal Petri nets are efficient that justifies their application as models of high performance computations and further development of Petri net based paradigm of computation [10].

Acknowledgement

The author would like to thank: Erik Winfree whose MCU2013 (http://mcu2013.ini.uzh.ch) invited talk inspired author to remember and apply early studied class of timed Petri nets with multichannel transitions; Turlough Neary, Damien Woods and Kenichi Morita for thier valuable comments on the initial idea.

References

[1] Cook, M.: Universality in elementary cellular automata. Complex Systems 15(1) (2004), 140. Available at http://www.complex-systems.com/pdf/15-1-1.pdf
[2] Cook, M.: A Concrete View of Rule 110 Computation. In T. Neary, D. Woods, A.K. Seda and N. Murphy (Eds.): The Complexity of Simple Programs 2008. EPTCS 1, 2009, pp. 3155, doi:10.4204/EPTCS.1.4
[3] Neary, T.: Small universal Turing machines. PhD thesis, Department of Computer Science, National University of Ireland, Maynooth (2008). Available at http://www.ini.uzh.ch/~tneary/tneary_Thesis.pdf
[4] Neary, T., Woods, D.: Four small universal Turing machines. Fundamenta Informaticae 91(1) (2009), 123-144, doi:10.3233/FI-2009-0036
[5] Neary, T., Woods D.: Small weakly universal Turing machines. 17th International Symposium on Fundamentals of Computation Theory (FCT 2009), volume 5699 of LNCS, pages 262-273, Wroclaw, Poland Sept. 2009. Available at http://www.ini.uzh.ch/~tneary/NearyWoods_FCT2009.pdf
[6] Peterson, J.: Petri Net Theory and the Modelling of Systems, Prentice-Hall, 1981, 264 p.
[7] Zaitsev, D.A.: Solving operative management tasks of a discrete manufacture via Petri net models. Ph.D. thesis, Kiev, the Academy of sciences of Ukraine, Institute of Cybernetics name of V.M.Glushkov, 1991, 124 p. In Russ.
[8] Zaitsev, D.A., Sleptsov A.I.: State equations and equivalent transformations for timed Petri nets. Cybernetics and Systems Analysis, Volume 33, Number 5 (1997), 659-672, doi:10.1007/BF02667189
[9] Zaitsev, D.A.: Universal Petri net. Cybernetics and Systems Analysis, Vol. 48, No. 4 (2012), 498-511, doi:10.1007/s10559-012-9429-4
[10] Zaitsev, D.A.: Petri Net Paradigm of Computation. In Book of abstracts of the International scientific conference on Computer Algebra and Information Technology, Odessa: ONU, August 20-26, 2012, pp. 107-114.
[11] Zaitsev, D.A.: Toward the Minimal Universal Petri Net. IEEE Transactions on Systems, Man, and Cybernetics: Systems, 2013, 1-12, doi:10.1109/TSMC.2012.2237549
[12] Zaitsev, D.A.: A Small Universal Petri Net. In Turlough Neary and Matthew Cook (Eds.): Proceedings Machines, Computations and Universality 2013 (MCU 2013), Zurich, Switzerland, 9/09/2013 - 11/09/2013, Electronic Proceedings in Theoretical Computer Science 128, pp. 190-202, doi:10.4204/EPTCS.128.22