Abstract

D.A. Zaitsev Constructing Universal Petri Net // Proceedings of VI report scientific-practical conference of the faculty and students, International Humanitarian University, Chair of Computer Engineering, Odessa (Ukraine), May 14-16, 2010, p. 27-32.

The universal inhibitor Petri net was constructed that executes an arbitrary given inhibitor Petri net. The inhibitor Petri net graph, its marking and the transitions firing sequence were encoded as 10 scalar nonnegative integer numbers and represented by corresponding places of universal net. The algorithm of inhibitor net executing that uses scalar variables only was constructed on its state equation and encoded by universal inhibitor Petri net. Subnets which implement arithmetic, comparison and copying operations were employed.

Keywords: *universal inhibitor Petri net, universal Turing machine, algorithm, encoding, control flow*

Д.А. Зайцев Построение универсальной сети Петри // Материалы VI отчетной научно-практической конференции профессорско-преподавательского состава и студенчества, Международный гуманитарный университет, кафедра компьютерной инженерии, Одесса, 14-16 мая, 2010, с. 27-32.

Построена универсальная ингибиторная сеть Петри, которая исполняет произвольную заданную ингибиторную сеть Петри. Граф ингибиторной сети Петри, ее маркировка и последовательность срабатывания переходов зашифрованы как 10 неотрицательных целых скалярных переменных, представленные соответствующими позициями универсальной сети. По уравнению состояний построен алгоритм исполнения ингибиторной сети, который использует только указанные скалярные переменные; алгоритм закодирован ингибиторной сетью Петри. Использованы подсети, которые реализуют арифметические и логические операции, копирование значений переменных.

Ключевые слова: *универсальная ингибиторная сеть Петри, универсальная машина Тюринга, алгоритм, шифрование, поток управления*

Д.А. Зайцев Побудова універсальної сіті Петрі // Матеріали VI звітної науково-практичної конференції професорсько-викладацького складу та студентства, Міжнародний гуманітарний університет, кафедра комп'ютерної інженерії, Одеса, 14-16 травня, 2010, с. 27-32.

Побудовано універсальну інгібіторну сіть Петрі, яка виконує довільну задану інгібіторну сіть Петрі. Граф інгібіторної сіті Петрі, її маркірування і послідовність спрацьовування переходів зашифровано як 10 невід'ємних цілих скалярних змінних поданих відповідними позиціями універсальної сіті. За рівнянням станів побудовано алгоритм виконання інгібіторної сіті, якій використовує лише вказані скалярні змінні; алгоритм закодовано інгібіторною сіттю Петрі. Використано підсіті, які реалізують арифметичні і логічні операції, копіювання значень змінних.

Ключеві слова: *універсальна інгібіторная сіть Петрі, універсальна машина Тюрінга, алгоритм, шифрування, потік управління*

Dmitry Zaitsev: http://member.acm.org/~daze
International Humanitarian University: http://www.mgu.com.ua

Zaitsev D.A.

Constructing Universal Petri Net*

There are known examples of constructing Universal Turing Machine [1]. It was shown that inhibitor Petri net is a universal algorithmic system [2,3]. The goal of the present paper is the constructing Universal Inhibitor Petri Net which executes any arbitrary given inhibitor Petri net.

The overall scope of the paper is the following: the dynamics of inhibitor net is described by the state equation; rules of inhibitor Petri net encoding are introduced for its representation by a constant number of scalar variables (places of the universal net); on the state equation, the algorithm of universal net executing is constructed with the application of the obtained inhibitor net encoding; then the algorithm is encoded by the inhibitor net UIPN which is a universal net.

Inhibitor Petri net is given by a pair of numbers, a pair of matrices and a vector: N=(m, n, B, D, Q_0), where m, n – numbers of places and transitions, B, D – matrices of the transitions input and output arcs multiplicity correspondingly, Q_0 – initial marking. The state equation (1) describes the dynamics of the net:

$$\begin{cases} q_j^k = q_j^{k-1} - x\left(b_{l,j}\right) + d_{l,j}, j = \overline{1,m} \\ u(t_i) = \bigwedge\limits_{j=1,m} ((y\left(b_{i,j}\right) \wedge (q_j^{k-1} = 0)) \vee (\bar{y}(b_{i,j}) \wedge (q_j^{k-1} \geq b_{i,j})), i = \overline{1,n} \\ u(t_l) = 1, l \in \overline{1,n} \\ k = 1,2, \dots \\ x(b) = \begin{cases} b, b \geq 0 \\ 0, b = -1 \end{cases}, y(b) = \begin{cases} 0, b \geq 0 \\ 1, b = -1 \end{cases}. \end{cases} \quad (1)$$

The code of a vector is a form of the numbers representation in a positional numerical system with the radix r and is given by the recurrent expression (2); for the decoding, (3) is applied. At the matrices encoding their vector representation with the expansion on lines is employed.

$$s_j = s_{j-1} \cdot r + q_{m-1-j}, s_0 = q_{m-1}, j = \overline{1, m-1}, r = \max_j q_j + 1. \quad (2)$$

$$q_j = s_{m-1-j} \bmod r, s_{m-1-(j+1)} = s_{m-1-j} \operatorname{div} r, s_{m-1} = s, \quad j = \overline{0, m-1}. \quad (3)$$

The code of a Petri net is represented by 10 places of UIPN with the following names m, n, sB, rB, sD, rD, sQ, rQ, k, sZ, where the prefix s denotes the code (2), the prefix r – applied radix, symbol Z denotes the transitions firing sequence, k – the length of the sequence. For the vector $\bar{u}$ encoding, the rules of a vector encoding (2) are employed with $r = 2$.

On the system (1), according to the described way of the encoding, the algorithm of inhibitor Petri net executing `AUIPN` is constructed in C-like pseudo language; the notations: `u` – the code of the firable transitions indicator, `l` – the number of the firing transition, `k` – the current step number; `CheckFire` – the check up of the transitions firing conditions, `PickFire` – the choice of the firing transition, `Fire` – the firing of transition; `NonDeterministic` – the nondeterministic

* A keynote speech to the Conference of International Humanitarian University, Odessa, Ukraine, May 14, 2010 (translated from Russian).

choice of number from the set {0,1}; the procedures MUL_ADD, MOD_DIV implement recurrent encoding/decoding according to (2), (3).

```
void AUIPN()
{
  uint u, l;
  inputXIPN();
  k=1; sZ=0;
  while(NonDeterministic())
  {
      CheckFire(&u);
      if(u==0) goto out;
      PickFire(u, &l);
      Fire(l);
      MUL_ADD(&sZ,n,l-1);
      k++;
  }
  out: outputXIPN();
}
void MUL_ADD(&x,y,z)
{
      (*x) = (*x) * y + z;
}
void MOD_DIV(&m,&x,y)
{
      (*m) = (*x) mod y;
      (*x) = (*x) div y;
}
void CheckFire(uint *u)
{
  uint i, j, qj, bij, ui, uij;
  uint sB1, sQ1;
  sB1=sB; &u=0;
  for(i=n; i>0; i--)
  {
      sQ1=sQ;
      ui=1;
      for(j=m; j>0; j--)
      {
        MOD_DIV(&qj,&sQ1,rQ);
        MOD_DIV(&bij,&sB1,rB);
        uij=1;
        if(bij==0) continue;
        bij--;
        if(bij==0) uij=(qj==0);
        else uij=(qj>=bij);
        ui=ui && uij;
      }
      MUL_ADD(&u,2,ui);
  }
}
```

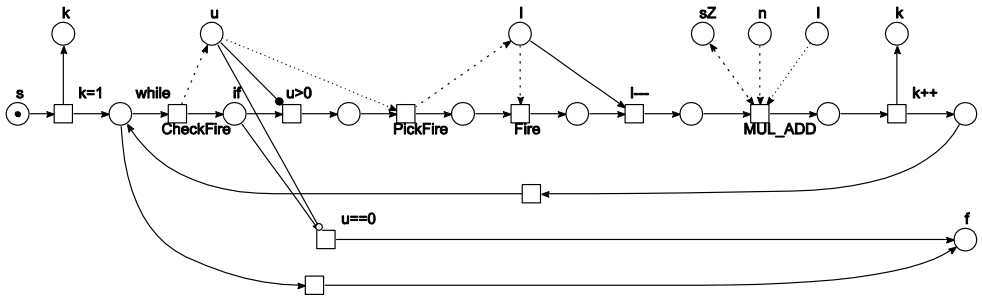
```
void PickFire(uint u, uint *l)
{
  uint ui, i;
  i=0;

  while(u>0)
  {
      MOD_DIV(&ui,&u,2);
      i++;
      if(ui==0) continue;
      if(NonDeterministic()) goto out;
  }
  out: *l=i;
}

void Fire(uint l)
{
  uint rQ1,maxQ1,shift,qj,bij,dij,j;
  uint sB1, sD1, sQ1;

  sB1=sB;sD1=sD;
  sQ1=0;rQ1=rQ+rD-1; maxQ1=0;

  shift=(n-l)*m;
  while(shift--)
  {
      MOD_DIV(&b,&sB1,rB);
      MOD_DIV(&d,&sD1,rB);
  }

  for(j=m; j>0; j--)
  {
      MOD_DIV(&qj,&sQ,rQ);
      MOD_DIV(&bij,&sB1,rB);
      if(bij>0) bij--;
      MOD_DIV(&dij,&sD1,rD);
      qj=qj-bij+dij;
      maxQ1=MAX(qj,maxQ1);
      MUL_ADD(&sQ1,rQ1,qj);
  }
  sQ=0; rQ=maxQ1+1;

  for(j=m; j>0; j--)
  {
      MOD_DIV(&qj,&sQ1,rQ1);
      MUL_ADD(&sQ,rQ,qj);
  }
}
```

Theorem 1. The algorithm AUIPN implements the dynamics of an arbitrary given inhibitor Petri net.

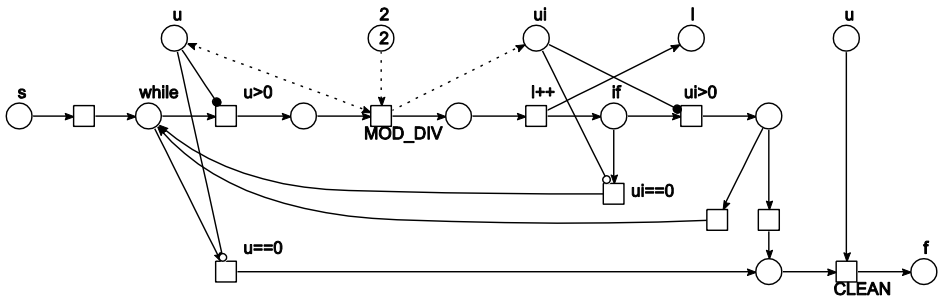
Theorem 2. The algorithm AUIPN can be represented by inhibitor Petri net.

The algorithm AUIPN was encoded by inhibitor Petri net UIPN (fig. 1).

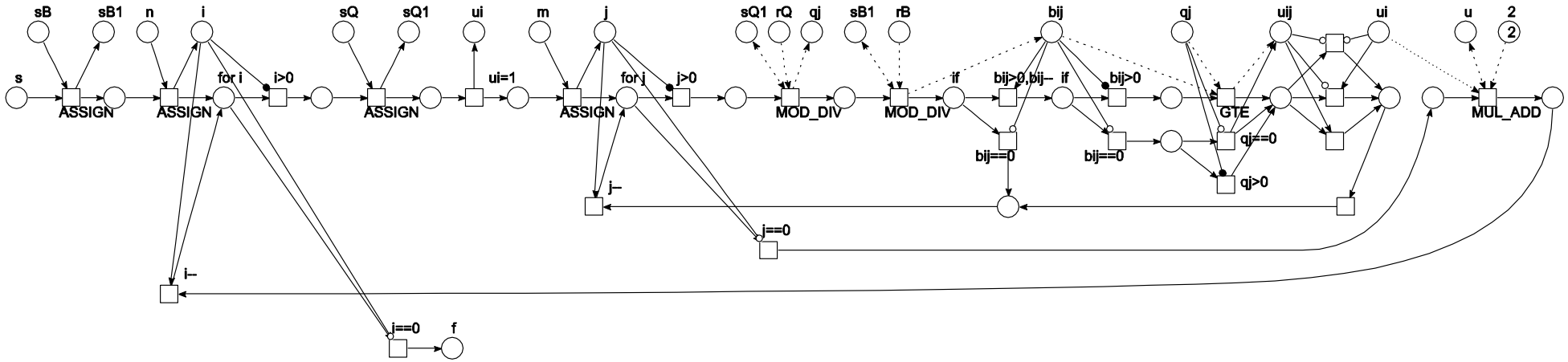
Theorem 3. The net UIPN is a universal inhibitor Petri net.



a) UIPN

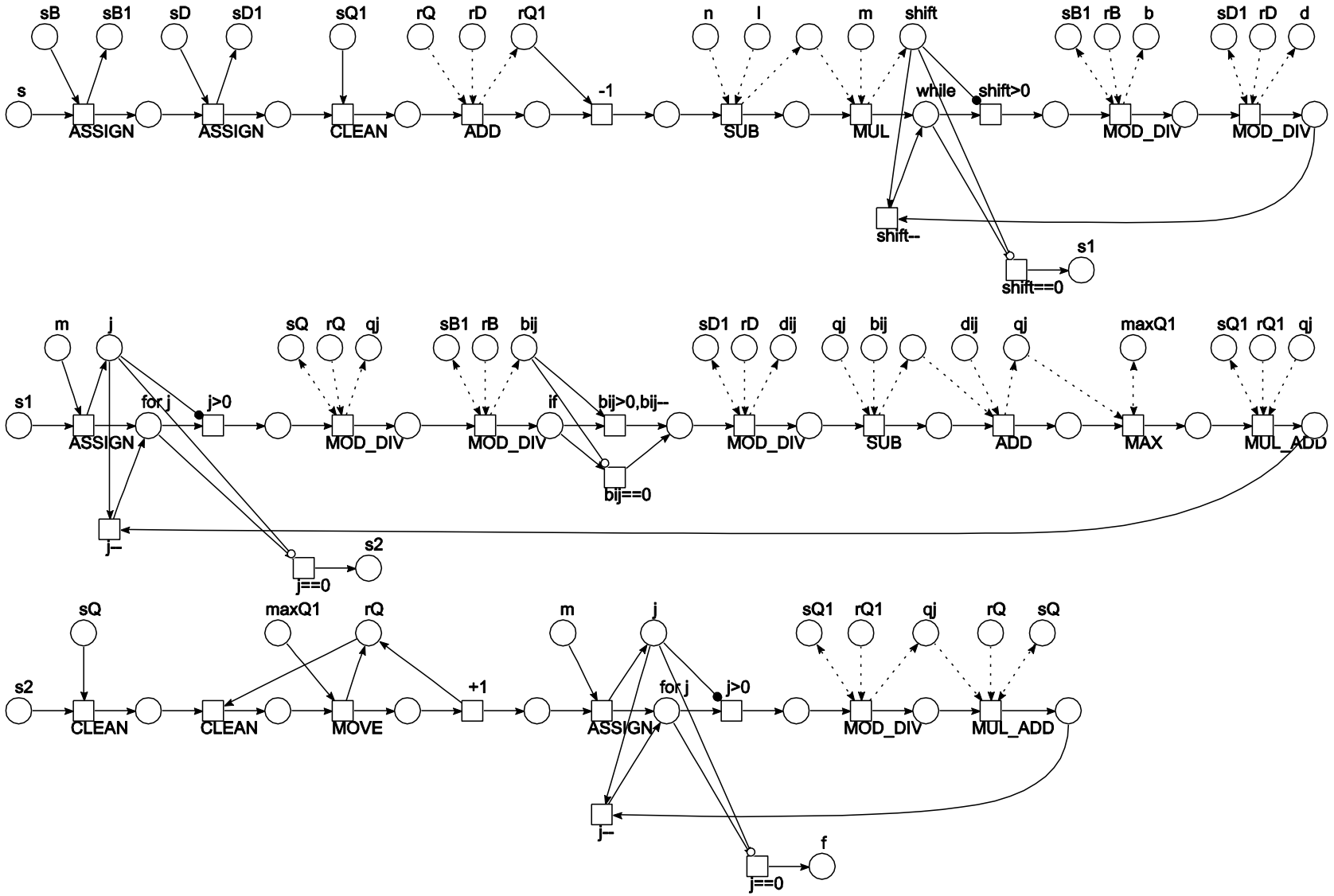


b) PickFire



c) CheckFire

Fig. 1. Universal inhibitor Petri net UIPN (beginning)



d) Fire

Fig. 1. Universal inhibitor Petri net UIPN (completion)

There were used auxiliary nets which implement arithmetic, logic operations and the rules of the variables values copying into the places of the universal net: CLEAN – cleaning, COPY – copying, MOVE – moving, ASSIGN – assigning, ADD – addition, SUB – substraction, MUL – multiplication, GTE – comparing, MAX – maximum, MUL_ADD – addition to the code, MOD_DIV – extraction from the code as well as the rules of sequential, branching, cycling algorithms encoding.

Thus, universal inhibitor Petri net was constructed; the direction for further work is the constructing of universal net with the minimal number of places (transitions), minimal value of employed marking.

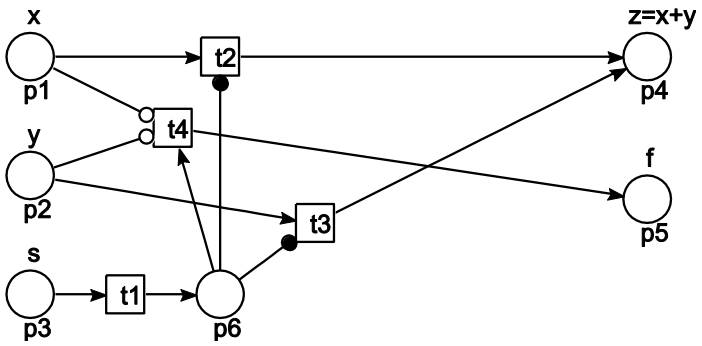
References

1. *The Universal Turing Machine. A Half-Century Survey / Rolf Herken (ed.), Springer-Verlag, Wien New York, 1994, 609 p.*
2. *Agerwala T. A Complete Model for Representing the Coordination of Asynchronous Processes, Baltimore, John Hopkins University, Hopkins Computer Science Program, Res. Rep. No. 32, July 1974.*
3. *Kotov V.E. Petri Nets – Moscow: Nauka, 1984. – 160 p. In Russ.*

Appendix: The examples of auxiliary nets

1) Addition ($z = x + y$)

ADD::



2) Addition to the code ($z = x \cdot y + v$)

MUL_ADD::