К вопросу о вычислительной сложности метода Тудика

Д.А.Зайцев

Аннотация. Выполнена оценка временной и емкостной сложности метода Тудика, предназначенного для решения однородных систем линейных диофантовых уравнений в целых неотрицательных числах. Определены условия полиномиальной сложности метода. Показано, что в худшем случае вычислительная сложность метода экспоненциальна по отношению к размерности системы. Установлено, что для разрежённых матриц имеет место эффект заливки, приводящий к оценкам сложности для худшего случая.

Ключевые слова: линейная система; диофантовы уравнения; неотрицательные решения; метод Тудика; вычислительная сложность

Метод Тудика [1] предназначен для решения однородных систем линейных диофантовых уравнений в целых неотрицательных числах. Такая на первый взгляд специфическая задача является широко распространённой в компьютерных науках, в особенности в теории сетей Петри. Инварианты сетей Петри [2] представляют собой целые неотрицательные решения линейных систем и применяются при структурном анализе сетей. Они позволяют определять такие важные свойства сети Петри как ограниченность, консервативность, повторяемость, живость.

Популярность метода Тудика во многом обусловлена простотой его описания. Последовательность элементарных преобразований матриц приводит к построению базисных решений. Поэтому, несмотря на упоминания об его асимптотически экспоненциальной сложности [3], метод применён в многочисленных компьютерных системах автоматизированного анализа моделей Петри [3,4]. Кроме того, в работе [5] представлено теоретическое обоснование метода, ранее характеризуемого как эвристического [3].

Следует обратить внимание на сложности, возникающие при решении линейных систем на множестве целых неотрицательных чисел. Указанное множество обладает лишь алгебраической структурой моноида, более бедной по сравнения с полями и кольцами, для которых решение линейной системы – это классическая задача [6].

Целью настоящей работы является построение точных оценок вычислительной сложности метода Тудика, определение условий, при которых метод может быть эффективно применён для анализа свойств сетей Петри.

Заметим, что проектирование интеллектуальных систем является одной из наиболее перспективных областей приложения теории сетей Петри. Инварианты сетей Петри традиционно используют для автоматического доказательства теорем в логическом программировании [7], для проверки непротиворечивости и полноты баз знаний [8].

1. Основные понятия и определения

Сеть Петри – это четвёрка $N = (P, T, F, W)$, где $P = \{p\}$ – конечное множество вершин, называемых позициями, $T = \{t\}$ – конечное множество вершин, называемых переходами, отношение смежности вершин $F = P \times T \bigcup T \times P$ задаёт множество дуг, соединяющих позиции и переходы. Таким образом, сеть Петри представляет собой двудольный ориентированный граф, одну долю вершин которого составляют позиции, а другую – переходы. Кроме того, рассматривают *кратности дуг сети*, представленные

отображением $W : F \to \mathrm{N}$. Кратность, отличную от единицы, указывают в виде числа на соответствующей дуге.

Как правило граф N дополняют функцией разметки, задающей первоначальное расположение фишек в позициях. Фишки представляют собой динамические элементы, которые перемещаются в результате срабатывания переходов. В настоящей работе рассматриваются структурные свойства сетей Петри, поэтому понятие динамики не вводится. Пример сети Петри, имеющей пять позиций и шесть переходов, представлен на Рис. 1.

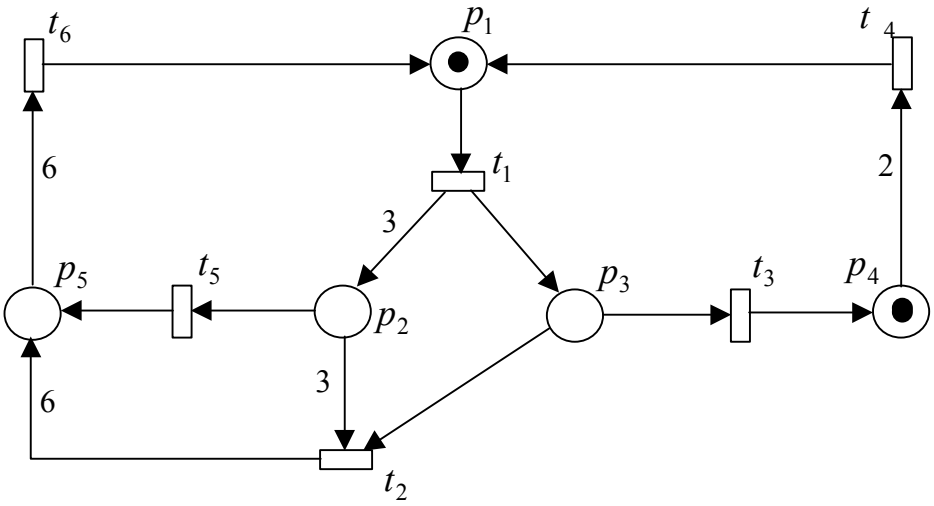

Рис. 1. Сеть Петри N_1.

Пусть $|P| = m$, $|T| = n$ и множества позиций и переходов занумерованы. Введём матрицы A^-, A^+ входящих и исходящих дуг переходов соответственно:

$$A^- = \left\| a^-_{i,j} \right\|,\ i = \overline{1,m},\ j = \overline{1,n};\ a^-_{i,j} = \begin{cases} w(p_i, t_j), & (p_i, t_j) \in F \\ 0, & \textit{иначе} \end{cases},$$

$$A^+ = \left\| a^+_{i,j} \right\|,\ i = \overline{1,m},\ j = \overline{1,n};\ a^+_{i,j} = \begin{cases} w(t_j, p_i), & (t_j, p_i) \in F \\ 0, & \textit{иначе} \end{cases}.$$

И, наконец, введём *матрицу инцидентности* A сети Петри как $A = A^+ - A^-$.

P-инвариантом сети Петри [1] называют целые неотрицательные решения системы

$$\bar{x} \cdot A = 0. \qquad (1)$$

T-инвариантом называют целые неотрицательные решения системы

$$\bar{y} \cdot A^T = 0.$$

Инварианты играют ключевую роль при исследовании таких свойств сетей как ограниченность, консервативность, живость [1].

2. Краткое описание метода Тудика

Метод Тудика [1] состоит в построении множества базисных решений системы (1) из единичной матрицы E следующим образом:

Шаг 0. Положим $D := A$, $R := E$.

Шаг 1. Если $D = 0$, то *Останов*. Матрица R является искомой матрицей базисных решений системы.

Шаг 2. Если все столбцы матрицы D содержат ненулевые коэффициенты одного знака, то *Останов*. Система имеет только тривиальное решение.

Шаг 3. Выберем столбец j матрицы D с минимальным значением произведения $\left|I^+\right| \times \left|I^-\right|$, где $I^+ = \{i \mid a_{i,j} > 0\}$, $I^0 = \{i \mid a_{i,j} = 0\}$, $I^- = \{i \mid a_{i,j} < 0\}$.

Шаг 4. Построим матрицу D' следующим образом: скопируем в матрицу D' строки I^0 матрицы D, затем допишем дополнительные строки для каждой комбинации $(k,r), k \in I^+, r \in I^-$, построенные как $\left|a_{r,j}\right| \cdot \bar{l}^k + \left|a_{k,j}\right| \cdot \bar{l}^r$.

Шаг 5. Аналогичные преобразования выполним над матрицей R для получения матрицы R'.

Шаг 6. Положим $D := D', R := R'$ и *перейдём к Шагу 1*.

Отметим, что на шаге 3 выбор очередного столбца обеспечивает минимальное количество новых решений соответствующего уравнения. Кроме того, строки матриц D и R могут быть сокращены совместно на общий делитель компонентов вектора на любом проходе алгоритма. Возможно также сокращение значений столбцов матрицы D.

Теоретическое обоснование метода представлено в работе [5]. Для генерации произвольного частного решения из базисных, представленных строками матрицы R, линейная комбинация с целыми неотрицательными коэффициентами расширена дополнительной операцией сокращения на общий делитель компонент вектора. Показано, что существуют частные решения, которые не могут быть получены без использования операции сокращения.

3. Пример решения системы

Рассмотрим применение метода для вычисления t-инвариантов сети N_1, представленной на Рис. 1. Необходимо решить систему уравнений

$$\bar{y} \cdot A^T = 0,\ A^T = \begin{vmatrix} -1 & 3 & 1 & 0 & 0 \\ 0 & -3 & -1 & 0 & 6 \\ 0 & 0 & -1 & 1 & 0 \\ 1 & 0 & 0 & -2 & 0 \\ 0 & -1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & -6 \end{vmatrix}.$$

Далее будем записывать пару матриц $\|D\|R\|$ на каждом проходе алгоритма:

1) $\left|\begin{array}{ccccc|c} -1 & 3 & 1 & 0 & 0 & 1\ 0\ 0\ 0\ 0\ 0 \\ 0 & -3 & -1 & 0 & 6 & 0\ 1\ 0\ 0\ 0\ 0 \\ 0 & 0 & -1 & 1 & 0 & 0\ 0\ 1\ 0\ 0\ 0 \\ 1 & 0 & 0 & -2 & 0 & 0\ 0\ 0\ 1\ 0\ 0 \\ 0 & -1 & 0 & 0 & 1 & 0\ 0\ 0\ 0\ 1\ 0 \\ 1 & 0 & 0 & 0 & -6 & 0\ 0\ 0\ 0\ 0\ 1 \end{array}\right|$, $j = 4$, $I^- = \{4\}$, $I^+ = \{3\}$

2) $$\left|\begin{array}{cccc||cccccc} -1 & 3 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & -3 & -1 & 6 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & -6 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & -2 & 0 & 0 & 0 & 2 & 1 & 0 & 0 \end{array}\right|, \; j=1, \; I^- = \{1\}, \; I^+ = \{4,5\}$$

3) $$\left|\begin{array}{ccc||cccccc} -3 & -1 & 6 & 0 & 1 & 0 & 0 & 0 & 0 \\ -1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 3 & 1 & -6 & 1 & 0 & 0 & 0 & 0 & 1 \\ 3 & -1 & 0 & 1 & 0 & 2 & 1 & 0 & 0 \end{array}\right|, \; j=2, \; I^- = \{1,4\}, \; I^+ = \{3\}$$

4) $$\left|\begin{array}{cc||cccccc} -1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 \\ 6 & -6 & 2 & 0 & 2 & 1 & 0 & 1 \end{array}\right|, \; j=1, \; I^- = \{1\}, \; I^+ = \{3\}$$

5) $$\left|\begin{array}{c||cccccc} 0 & 1 & 1 & 0 & 0 & 0 & 1 \\ 0 & 2 & 0 & 2 & 1 & 6 & 1 \end{array}\right|.$$

Таким образом, получено два базисных решения. Следовательно, сеть Петри N_1 является t-инвариантной.

4. Условия полиномиальной сложности

Следует отметить, что в системе (1) столбцы матрицы A представляют уравнения, а строки соответствуют переменным. На каждом проходе алгоритма обнуляется один из столбцов, при этом образуются новые переменные. В заключение мы приходим к нулевой матрице D и новые переменные, таким образом, становятся свободными.

Следовательно, количество проходов алгоритма не превышает n. Основными источниками роста вычислительной сложности выступают, во-первых, увеличения числа строк матрицы, а во-вторых, рост абсолютного значения элементов.

Обозначим q максимальное количество переходов либо позиций сети $q = \max(|P|,|T|)$, а $m(i)$ – количество строк матрицы D на i-м проходе алгоритма. Тогда имеем следующее рекуррентное соотношение:

$$\begin{cases} m(i+1) = m(i) - (|I^-| + |I^+|) + |I^-| \cdot |I^+|, \\ m(0) = q. \end{cases} \qquad (2)$$

Действительно, на очередном проходе $|I^-| + |I^+|$ строк матрицы заменяются новыми $|I^-| \cdot |I^+|$ строками. Знак выражения $h(i) = |I^-| \cdot |I^+| - |I^-| + |I^+|$ определяет, увеличивается, либо уменьшается количество строк матрицы. Выполним оценку сложности алгоритма в лучшем и в худшем случаях.

В лучшем случае $h(i) \le 0$. То есть

$$|I^-| + |I^+| \ge |I^-| \cdot |I^+|. \qquad (3)$$

Тогда сложность алгоритма полиномиальна и не превышает $o(q^3)$, так как количество строк матрицы не возрастает и на каждом проходе выполняется около q^2 операций.

Рассмотрим сочетание значений $\left|I^-\right|$ и $\left|I^+\right|$, обеспечивающее выполнение неравенства (3). Возможно два варианта:

1. $\left|I^-\right|=1$ либо $\left|I^+\right|=1$.
2. $\left|I^-\right|=\left|I^+\right|=2$.

Таким образом, каждый из переходов имеет не более четырёх инцидентных дуг, либо имеет ровно одну входящую (одну исходящую) дугу. Причём эти ограничения должны оставаться справедливыми в процессе выполнения алгоритма.

5. Сложность в худшем случае

В худшем случае $\left|I^+\right|=\left|I^-\right|=\dfrac{m(i)}{2}$, так как при таком соотношении размеров множеств достигается максимум функции $h(i)$ на текущем проходе алгоритма, ведь $\left|I^+\right|+\left|I^-\right|\le m(i)$. Таким образом, из (2) получаем:

$$\begin{cases} m(i+1)=\left(\dfrac{m(i)}{2}\right)^2, \\ m(0)=n. \end{cases} \qquad (4)$$

Например: $m(1)=\dfrac{n^2}{2^2}$, $m(2)=\dfrac{n^4}{2^6}$, $m(3)=\dfrac{n^8}{2^{14}}$,…

Лемма 1. Рекуррентное выражение (4) можно представить в явной форме записи как:

$$m(i)=\frac{n^{2^i}}{2^{2^{i+1}-2}}. \qquad (5)$$

Доказательство. Построим доказательство на основе индукции. Предположим, что выражение (5) представляет рекуррентное соотношение (4) для $i=1,2,...,j$. Пусть

$$m(j)=\frac{n^{2^j}}{2^{2^{j+1}-2}}.$$

Докажем справедливость утверждения леммы для шага $j+1$. В соответствии с (4):

$$m(j+1)=\left(\frac{m(j)}{2}\right)^2=\left(\frac{n^{2^j}}{2^{2^{j+1}-2}}\Big/2\right)^2=\frac{\left(n^{2^j}\right)^2}{\left(2^{2^{j+1}-2}\right)^2\cdot 2^2}=\frac{n^{2^j\cdot 2}}{2^{(2^{j+1}-2)2+2}}=\frac{n^{2^{j+1}\cdot}}{2^{2^{j+2}-2}}.$$

□

Тогда сложность алгоритма как количество выполняемых операций может быть оценена следующим выражением:

$$Y_T(q)=\sum_{i=i,n} n\cdot\frac{n^{2^i}}{2^{2^{i+1}-2}}\sim q^2\cdot\frac{q^{2^q}}{2^{2^{q+1}}}\sim\frac{q^{2^q}}{2^{2^{q+1}}}.$$

Представим

$$q^{2^q}=\left(2^{\log_2 q}\right)^{2^q}=2^{2^q\cdot\log_2 q}=\left(2^{2^q}\right)^{\log_2 q}.$$

А также

$$2^{2^{q+1}}=2^{2^q\cdot 2}=\left(2^{2^q}\right)^2.$$

Тогда

$$Y_T(q) = \left(2^{2^q}\right)^{\log_2 q} \Big/ \left(2^{2^q}\right)^2 = \left(2^{2^q}\right)^{\log_2 q - 2} = 2^{2^q \cdot (\log_2 q - 2)} \quad . \qquad (6)$$

Полученный результат можно представить в виде следующей теоремы.

Теорема 1. Временная сложность метода Тудика не превышает величины $Y_T(q)$, заданной выражением (6), где q - количество вершин сети.

Таким образом, метод Тудика имеет асимптотически экспоненциальную временную сложность. Представленное в (6) значение $Y_T(q)$ действительно растёт довольно быстро с ростом q. Например: $Y_T(5) \approx 10^9$, $Y_T(10) \approx 10^{300}$. Таким образом, даже для небольших сетей вычисления могут потребовать тысячелетий.

Оценим емкостную сложность метода. Пусть a – максимальное по модулю значение числа в матрице A: $a = \max\limits_{i,j} \left| a_{i,j} \right|$. Оценим значение максимального по модулю числа в матрицах D, R на i-м проходе алгоритма, обозначенное как $v(i)$. В соответствии с описанием шага 4 алгоритма максимальное значение числа на предыдущем и последующем проходах алгоритма связаны рекуррентным следующим соотношением:

$$\begin{cases} v(i+1) = 2 \cdot \left(v(i)\right)^2, \\ v(0) = a. \end{cases} \qquad (7)$$

Лемма 2. Рекуррентное выражение (7) можно представить в явной форме записи как:

$$v(i) = 2^{2^{i+1}-1} \cdot l^{2^i} \quad . \qquad (8)$$

Также как в лемме 1 доказательство может быть выполнено индуктивно.

Обозначим $k = \log_2 a$. Тогда (8) представимо в виде:

$$v(i) = 2^{2^{i+1}-1} \cdot \left(2^k\right)^{2^i} = 2^{2^{i+1}-1} \cdot 2^{k \cdot 2^i} = 2^{2^{i+1} + k \cdot 2^i - 1} = 2^{(k+2)\cdot 2^i - 1}.$$

Так как $v(i)$ возрастает с ростом i, наибольшее значение достигается на проходе с номером n:

$$v(n) = 2^{(k+2)\cdot 2^n - 1}.$$

Тогда для хранения числа потребуется не менее $b(n)$ бит, где

$$b(n) = \log_2 v(n) = \left(k+2\right) \cdot 2^n - 1 \sim \left(\log_2 a + 2\right) \cdot 2^n .$$

Количество строк матриц на заключительном проходе алгоритма представлено выражением (6). Тогда емкостную сложность метода можно оценить следующим образом:

$$Z_T(q) = b(q) \cdot Y_T(q) \cdot n \approx 2^{2^q \cdot (\log_2 q - 2)} \cdot \left(\log_2 a + 2\right) \cdot 2^q = \left(\log_2 a + 2\right) \cdot 2^{2^q \cdot (\log_2 q - 2) + q} . \qquad (9)$$

Полученный результат можно представить в виде следующей теоремы.

Теорема 2. Емкостная сложность метода Тудика не превышает величины $Z_T(q)$, заданной выражением (9), где q – количество вершин сети.

Таким образом, получены оценки временной и емкостной сложности метода Тудика в худшем случае. Оценки являются экспоненциальными, что затрудняет практическое применение метода при исследовании сетей большой размерности.

6. Оценка сложности для разрежённых матриц

Сложность в худшем случае оценивалась для матрицы A, большинство значений которой являются ненулевыми. В сетях Петри, моделирующих реальные объекты, связи вершин локализованы. Количество инцидентных вершин, как правило не превышает некоторой заранее заданной константы, значение которой во много раз меньше размерности сети. Таким образом, исходная матрица является разрежённой.

Можно предположить, что оценки сложности метода Тудика для разрежённых матриц будут гораздо более оптимистичными, чем полученные в предыдущем разделе. Тем более, что в литературе имеются сообщения об успешном практическом применении метода для сетей имеющих несколько сот вершин [4].

Итак, пусть, каждая из вершин сети Петри имеет не более k входящих и не более k исходящих дуг. Таким образом, каждый столбец и каждая строка матрицы A содержит не более $2 \cdot k$ ненулевых элементов, причём k из них отрицательны, а остальные k положительны. Рассмотрим каким образом будет изменятся заполненность матриц D, R ненулевыми элементами, так как именно она существенно влияет на оценки сложности алгоритма.

С одной стороны, на каждом проходе обнуляется столбец матрицы D; таким образом, количество гарантированно нулевых элементов в строках возрастает. С другой стороны, при образовании новой строки матрицы D на шаге 4 суммируются значения двух строк. При суммировании строк количество ненулевых элементов $u(i)$ в худшем случае удваивается:

$$\begin{cases} u(i+1) = 2 \cdot v(i), \\ u(0) = 2 \cdot k. \end{cases}$$

Тогда

$$u(i) = 2^{i+1} \cdot k \,. \qquad (10)$$

Темпы заполнения строки ненулевыми элементами, представленные экспоненциальной функцией (10), значительно опережают темпы обнуления столбцов, характеризуемые линейной функцией $f(i) = i$.

Таким образом, уже на проходе r новые строки матрицы будут заполнены ненулевыми элементами, где значение r задаётся неравенством

$$u(r) \geq n - r \,.$$

Для простоты оценки, рассмотрим заполнение всей строки: $2^{r+1} \cdot k \geq n$; тогда $2^r \geq \dfrac{n}{2 \cdot k}$ или:

$$r \geq \log_2 \frac{n}{2 \cdot k} \,.$$

При этом на проходе r появляется не менее k^2 заполненных строк и дальнейшие оценки сложности соответствуют оценкам (6), (9), выполненным для худшего случая.

Так, например при $n = 1000$ и $k = 5$ получаем $r \geq 7$. То есть после седьмого прохода новые строки матрицы D будут заполнены ненулевыми элементами.

Таким образом, обнаружен эффект, состоящий в экспансии ненулевых элементов в процессе применения метода Тудика к разрежённой матрице и названный эффектом заливки матрицы. Полученные результаты позволяют сделать вывод, что реальная вычислительная сложность метода Тудика незначительно отличается от оценок, выполненных для худшего случая и является экспоненциальной.

Таким образом, получены оценки временной и емкостной сложности метода Тудика, предназначенного для решения линейных однородных систем диофантовых уравнений в целых неотрицательных числах. Определены условия полиномиальной сложности метода. Показано, что в общем случае метод Тудика имеет асимптотически экспоненциальную сложность, что затрудняет его практическое применение. Кроме того, оценена сложность метода для разрежённых матриц. Показано, что имеет место эффект заливки матрицы, в результате которого разрежённая матрица на первых нескольких шагах алгоритма заполняется ненулевыми значениями и дальнейшие оценки сложности соответствуют оценкам, выполненным для худшего случая.

Литература

1. Toudic J.M. Linear Algebra Algorithms for the Structural Analysis of Petri Nets // Rev.Tech. Thomson CSF, Vol. 14, No. 1, 1982, pp. 136-156.
2. Murata T. Petri Nets: Properties, Analysis and Applications // Proc. IEEE, Vol.77, No. 4, 1989, pp. 541-580.
3. Бандман М.К., Бандман О.Л., Есикова Т.Н. Территориально-производственные комплексы: Прогнозирование процесса формирования с использованием сетей Петри.- Новосибирск. Наука, 1990.- 303с.
4. Бандман О.Л. Поведенческие свойства сетей Петри / Известия АН СССР. Техническая кибернетика. - 1987, № 5, с. 134-150.
5. Zaitsev D.A. Formal Grounding of Toudic Method // Proceedings of the 10th Workshop "Algorithms and Tools for Petri Nets".- Eichstaett, Germany, September 26-27, 2003, pp. 184-190.
6. B.L.Van Der Warden, Algebra, Berlin, Heidelberg, Springer-Verlag, 1971.
7. Peterke G., Murata T. Proof procedure and answer extraction in Petri net model of logic programs // IEEE Trans. Soft. Eng., no 2, 1989, Vol. 15, pp. 209-217.
8. Liu N.K., Dillon J. Detection of consistency and completeness in expert systems using numerical Petri nets // Joint Artif. Intell. Conf., Sydney, November 2-4, 1987, pp. 119-134.

Published: Artificial Intelligence, no. 1, 2004, p. 29-37. In Russ.