**УДК 004.822**

**doi.org/10.15407/fmmit2023.36.131**

# Модифікація алгоритму побудови мінімальної генеруючої множини розв'язків СЛР в множині натуральних чисел

Сергій Кривий[1], Олексій Чугаєнко[2]

[1] д. ф.-м. н., професор, Київський Національний університет імені Тараса Шевченка, 03680, Київ, проспект академіка Глушкова 4Д, e-mail: sl.krivoi@gmail.com

[2] старший програміст ТОВ «SAMSUNG RND Ukraina», 01032, Київ, в. Толстого, б. 57. email: firestream_13@yahoo.com

*У роботі наведена оптимізаційні перетворення алгоритму побудови мінімальної генеруючої множини розв'язків системи лінійних однорідних рівнянь в множині натуральних чисел, які покращують часові характеристики цього алгоритму з обгрунтуванням та ілюстрацією роботи на прикладах.*

**Ключові слова:** Діофантові рівняння, алгоритм, розв'язання, складність

**Вступ.** Існує багато задач зводиться до розв'язання систем лінійних рівнянь (СЛР) і тому було розроблено багато методів розв'язання СЛР. Особливе місце в цьому ряду займають СЛР, розв'язки яких шукаються в множині натуральних чисел $N$. Більше століття тому було встановлено, що СЛР такого типу мають скінченний базис множини всіх розв'язків [1,2], але алгоритму побудови цього базису не було (тоді і поняття алгоритму не існувало) Роботи по пошуку алгоритмів розв'язання СЛР велися весь час, але особлива активність в цьому напрямку почалася на початку 80-х років 20-го століття у зв'язку з роботами по побудові пруверів для теорій першого порядку. Побудова пруверів вимагала розв'язання проблеми уніфікації термів у асоціативно-комутативних теоріях першого порядку, яке зводилося до розв'язання СЛР в множині $N$. З'явилися алгоритми розв'язання одного лінійного однорідного рівняння (ЛОР) [3], систем ЛОР (СЛОР) [4,5,6] та інші. В кінці 20-го століття був розроблений TSS-метод і алгоритм розв'язання СЛР у множині $N$ для аналізу мереж Петрі [7]. Алгоритм, побудований на TSS-методі, описаний в монографії [8] і застосовний для розв'язання систем лінійних обмежень типу рівностей і нерівностей. Особливістю цього алгоритму є те, що в процесі пошуку розв'язків систем він не використовує операцію ділення, за винятком скорочення на найбільший спільний дільник. Алгоритм знаходить мінімальну генеруючу множину розв'язків СЛОР у множині $N$. Слід зазначити, що всі алгоритми розв'язання СЛОР і TSS-алгоритм в тому числі у множині $N$ досить складні [8, 9].

## 1. Алгоритм побудови мінімальної генеруючої множини розв'язків СЛОР

Нехай дана СЛОР з цілими коефіцієнтами

$$S = \begin{cases} L_1(x) = a_{11}x_1 + a_{12}x_2 + ... + a_{1q}x_q = 0, \\ ..... \\ L_p(x) = a_{p1}x_1 + a_{p2}x_2 + ... + a_{pq}x_q = 0, \end{cases} \quad (1)$$

де $a_{ij} \in Z$ (множина цілих чисел), а $x_{ij}$ приймають значення у множині $N$. Розглянемо множину векторів канонічного базису $M_0' = \{e_1,e_2,\ldots,e_q\}$ і перше рівняння $L_1(x) = a_{11}x_1+a_{12}x_2+...+a_{1q}x_q=0$ системи $S$. За допомогою функції $L_1(x)$ розіб'ємо елементи множини $M_0'$ на такі три групи: $M_1^0 = \{e \mid L_1(e) = 0\}$, $M_1^+ = \{e \mid L_1(e) > 0\}$ і $M_1^- = \{e \mid L_1(e) < 0\}$. Зрозуміло, що коли одна з множин $M_1^0 \cup M_1^+$ чи $M_1^0 \cup M_1^-$ пуста, то рівняння $L_1(x) = 0$ не має нетривіальних розв'язків у множині натуральних чисел. Якщо множини $M_1^0$, $M_1^+$, $M_1^-$ непусті, то розглянемо множину $M_1' = M_1^0 \cup \{y_{ij} \mid y_{ij} = -L_1(e_i)e_j + L_1(e_j)e_i\}$, де $e_j \in M_1^+$, $e_i \in M_1^-$. Використовуючи функцію $L_2(x)$, розіб'ємо елементи множини $M_1'$ аналогічно до попереднього також на три групи: $M_2^0 = \{e \mid L_2(e) = 0\}$, $M_2^+ = \{e \mid L_2(e) > 0\}$ і $M_2^- = \{e \mid L_2(e) < 0\}$. Припустимо, що принаймні дві з цих множин непусті, тоді побудуємо множину

$$M_2' = M_2^0 \cup \{y_{ij} \mid y_{ij} = -L_2(e_i)e_j + L_2(e_j)e_i, \text{ де } e_j \in M_2^+, e_i \in M_2^-\}.$$

Нехай таким способом побудовано множину $M_j'$ із множин $M_j^0 = \{e_r^0 \mid L_j(e_r^0) = 0\}$, $M_j^+ = \{e_i^+ \mid L_j(e_i^+) > 0\}$ й $M_j^- = \{e_s^- \mid L_j(e_s^-) < 0\}$ за допомогою функції $L_j(x)$ і ця множина непуста. Справедлива очевидна

**Теорема 1.** Елементи множини $M_j'$ – розв'язки системи рівнянь

$$L_1(x) = 0 \wedge L_2(x) = 0 \wedge \ldots \wedge L_j(x) = 0.$$

**Означення 1.** Побудована множина $M_j'$ називається зрізаною множиною розв'язків або TSS (від англійського *truncated set of solutions*). Ця назва себе виправдовує, оскільки отримана множина розв'язків не завжди буде базисом множини всіх розв'язків системи.

Генеруюча властивість TSS випливає з такої теореми.

**Теорема 2.** Нехай M – TSS системи лінійних однорідних рівнянь (1) у множині $N$, побудована вищеописаним способом, і M' множина всіх розв'язків цієї системи. Тоді довільний розв'язок $x \in M'$ представляється у вигляді лінійної комбінації $kx = a_1e_1 + a_2e_2 + \cdots + a_me_m$, де $k \geq 1$, $a_i \in N$, $e_i \in M$, $i = 1,...m$ [8].

Нехай $(x,y)$ означає скалярний добуток векторів $x$ і $y$, а $L_i(x)$ – ліву частину $i$-го рівняння $S$. З вищесказаного випливає такий алгоритм побудови TSS для СЛОДР у множині натуральних чисел $N$.

**TSSEQN**(M$p$,$q$)
**Вхід:** Матриця СЛОДР розмірності $p \times q$.
**Вихід:** TSS-множина M розв'язків СЛОДР або ``несумісна".
**Метод:**

```
M:={e1,…,eq | ei канонічний вектор N^q});
for i := 1 to p do
   M:=TSSEQN1(M,Li(x));
   if M=∅ then (print(несумісна);STOP) else CLEAN(M);
print(M);
```

**TSSEQN1**(M,$L(x)$)

```
begin
  M^0:= ∅; \ M^+:=∅; M:= ∅;
  forall e ∈ M do
    if L(e)=0 then M^0:=M^0 ∪ {e} else
      if L(e) > 0 then M^+:=M^+ ∪ {e} else M^-:=M^- ∪ {e};
      M':= M^0;
      if M^+ ≠ ∅ ∧ M^- ≠ ∅ then
        forall u ∈ M^+ do
          forall v ∈ M^- do e := -L(v)u + L(u)v; M':=M' ∪ {e} od;
  return(M');
end
```

Процедура CLEAN(M) виконує чистку отриманної множини розв'язків M. В загальному випадку така чистка зводиться до пошуку невід'ємних розв'язків СЛНР таких, що коли розв'язок $a$ є невід'ємною лінійною комбінацією решти розв'язків із множини M, то він із M вилучається.

З наведених теорем 1, 2 випливає, що кожний вектор TSS можна поділити на НСД його координат, якщо цей НСД відмінний від одиниці. Це дає змогу зменшити величину координат цих векторів і ефективніше проводити обчислення.

Розглянемо приклади, які ілюструють роботу алгоритму **TSSEQN** та властивості множини розв'язків, отриманих цим алгоритмом. Покажемо спочатку, що результати обчислень TSS залежать від порядку рівнянь у системі.

**Приклад 1.** Знайти TSS розв'язки у множині натуральних чисел $N$:

$$S=\begin{cases}-x_1+x_2+2x_3-3x_4=0,\\ -x_1+3x_2-2x_3-x_4=0.\end{cases}$$

*Розв'язання.* TSS першого рівняння набуває вигляду:

$e_1$=(1,1,0,0), $e_2$ = (2,0,1,0) $e_3$=(0,3,0,1), $e_4$=(0,0,3,2).

Підставляємо в друге рівняння знайдені розв'язки і знаходимо значення: 2, -4, 8, -8. Комбінуючи їх за знайденими значеннями, дістаємо таку TSS системи:

$s_1=2e_1+e_2$=(4,2,1,0), $s_2=2e_2+e_3$=(4,3,2,1), $s_3=4e_1+e_4$=(4,4,3,2), $s_4 = e_3+e_4$=(0,3,3,3),

Де останній розв'язок після скорочення на НСД координат набуває вигляду $s_4$ = =(0,1,1,1). Розв'язки $s_2$ і $s_3$ лінійно виражаються через $s_1$ і $s_4$ , дійсно

$s_2 = s_1 + s_4$ і $s_3 = s_1 + 2s_4$.

Отже, розв'язки $s_1$ і $s_4$ складають мінімальну породжуючи множину всіх розв'язків початкової системи.

А тепер почнемо будувати TSS з другого рівняння даної системи. Дістаємо таку TSS

$e_1$ = (3,1,0,0), $e_2$ = (0,2,3,0), $e_3$ = (0,1,0,3).

Підстановка в перше рівняння знайдених розв'язків другого рівняння дає значення -1,4, -4. Тоді TSS всієї системи маї вигляд:

$s_1=4e_1+e_2$=(12,6,3,0) (після скорочення на НСД) = (4,2,1,0),

$s_2=e_2+e_3$=(0,3,3,3) ( після скорочення на НСД) = (0,1,1,1).

TSS розв'язки для даної системи є базисом всієї множини розв'язків, але в загальному випадку мінімальна породжуючи множина розв'язків базисом не буде, як показує наступний приклад. Отже, другий спосіб побудови TSS не обчислює «лишніх» розв'язків СЛОР.

**Приклад 2.** Перевірити сумісність СЛОДР:

$$S=\begin{cases} L_1(x)=4x_1+2x_2-3x_3-2x_4-x_5=0, \\ L_2(x)=2x_1-x_2-4x_3+x_4+5x_5=0, \\ L_3(x)=0x_1+2x_2+4x_3+0x-9x_5=0. \end{cases}$$

TSS $M_1'$ складають вектори:$e_1$ = (3,0,4,0,0), $e_2$ = (0,3,2,0,0), $e_3$ = (1,0,0,2,0), $e_4$ = (0,1,0,1,0), $e_5$ = (1,0,0,0,4), $e_6$ = (0,1,0,0,2). Базис множини розв'язків рівняння $L_1(x) = 0$ (який можна знайти, наприклад, методом із роботи [3]) складають вектори: $e_1$ = (3,0,4,0,0), $e_2$ = (0,3,2,0,0), $e_3$ = (1,0,0,2,0), $e_4$ = (0,1,0,1,0), $e_5$ = (1,0,0,0,4), $e_6$ = (0,1,0,0,2), $e_7$ = (2,0,2,1,0), $e_8$ = (1,1,2,0,0), $e_9$ = (1,0,0,1,2), $e_{10}$ = (0,2,1,0,1), $e_{11}$ = (1,0,1,0,1). Вектори $e_7$ - $e_{11}$ цього базису мають такі розклади через вектори $e_1 – e_6$: $2e_7 = e_1 + e_3$, $3e_8 = e_1 + e_2$, $2e_9 = e_3 + e_5$, $2e_{10} = e_2 + e_6$, $4e_{11} = e_1 + e_5$.

Знаходячи значення функції $L_2(x)$ на векторах із $M_1'$, маємо
$L_2(e_1) = -10$, $L_2(e_2) = -11$, $L_2(e_3) = 4$, $L_2(e_4) = 0$, $L_2(e_5) = 22$, $L_2(e_6) = 9$. Будуємо $M_2'$:
$e_1'$= (0,1,0,1,0), $e_2'=2e_1+5e_3$=(11,0,8,10,0), $e_3'=4e_2+11e_3$=(11,12,8,22,0), $e_4'=2e_2+e_5$ =(1,6,4,0,4),
$e_5'=9e_2+11e_6$ = (0,19,9,0,11), $e_6'=11e_1+5e_5$ = (19,0,22,0,10), $e_7'= 9e_1 + 10e_6$ = (27,10,36,0,20).
Якщо одержану множину розв'язків проаналізувати на лінійну залежність, то дістаємо таку множину векторів $M_2'$ = {$e_1'$ = (0,1,0,1,0), $e_2'$ = (11,0,8,10,0), $e_4'$ = (0,19,9,0,11), $e_5'$ = =(19,0,22,0,10)}, оскільки $e_3' = e_2' + 12e_1'$, $19e_4'=6e_5' + e_6'$, $19e_7' = 10e_5' + 27e_6'$ і їх можна вилучити із TSS.

Базис множини розв'язків системи $L_1(x)=0 \wedge L_2(x)=0$ складають такі вектори:
$s_1$ = (0,1,0,1,0), $s_2$ = (11,0,8,10,0), $s_3$ = (16,1,19,0,9), $s_4$ = (0,19,9,0,11),
$s_5$ = (19,0,22,0,10), $s_6$ = (1,6,4,0,4), $s_7$ = (13,2,16,0,8), $s_8$ = (10,3,13,0,7),
$s_9$ = (7,4,10,0,6), $s_{10}$ = (4,5,7,0,5), $s_{11}$ = (3,0,3,1,1).
Вектори $s_6 – s_{11}$ і $s_3$ є лінійними комбінаціями векторів $s_1$=(0,1,0,1,0), $s_2$=(11,0,8,10,0), $s_4$=(0,19,9, 0,11), $s_5$ = (19,0,22,0,10), мають такий вигляд:
$19s_6 = 6s_4 + s_5$, $19s_7 = 2s_4 + 13s_5$, $19s_8 = 3s_3 + 10s_5$,
$19s_9 = 4s_4 + 7s_5$, $19s_{10} = 5s_4 + 4s$, $10s_{11} = s_2 + s_5$,$19s_3=s_4+16s_5$.
Значення $L_3(s_1) = 2$, $L_3(s_2) = 32$, $L_3(s_4) = -25$, $L_3(s_5) = -2$, тобто система сумісна і має принаймні чотири розв'язки:
$s_1'$=(19,1,22,1,10), $s_2'$= (275,608,488,250,352), $s_3'$=(0,63,18,25,22), $s_4'$ = (63,0,72,2,32).
Але розв'язки $s_1'$,$s_2'$ лінійно виражаються через $s_3'$=(0,63,18,25,22), $s_4'$ = (63,0,72,2,32). Справді, 63·(19,1,22,1,10) = 19·(63,0,72,2,32) + (0,63,18,25,22) і 63·(275,608,488,250,352) = =275·(63,0,72,2,32) + 608·(0,63,18,25,22). Отже, розв'язки $s_3'$=(0,63,18,25,22), $s_4'$ = (63,0,72,2,32) складають TSS системи $S$.

## 2. Оптимізаційні перетворення TSS-алгоритма

Тепер можна описати оптимізаційні перетворення та оцінити їх ефективність на конкретних задачах.

***Перше оптимізаційне перетворення*** випливає з аналізу ситуації в прикладі 1. Це перетворення зводиться до вибору першого рівняння системи, з якого починається побудова TSS: ***слід вибирати серед рівнянь системи те рівняння, у якого величина kn+m мінімальна***, де $k$ – кількість додатних коефіцієнтів, $n$ – кількість від'ємних коефіцієнтів і $m$ – кількість нульових коефіцієнтів у системі. Це перетворення ефективне, як показують експерименти, але воно не завжди можливе в ситуації, коли для всіх рівнянь системи величина $kn+m$ однакова.

***Друге оптимізаційне перетворення*** випливає з першого і зводиться до діагоналізації матриці системи. Діагоналізація має поліноміальну складність і дозволяє зменшити величину ***kn+m,*** про яку йшлося в першому перетворенні. Простий приклад ілюструє ефективність цього перетворення: нехай матриця $A$ СЛОДР має вигляд:

$$A=\begin{pmatrix} 2 & -1 & 3 & 1 & -4 & 2 \\ 1 & 0 & -2 & -3 & 2 & 1 \\ -3 & 1 & 1 & 0 & -1 & 2 \\ 4 & 1 & -1 & -2 & 0 & 1 \end{pmatrix}, \qquad A'=\begin{pmatrix} 0 & 0 & 0 & 31 & 5 & -38 \\ 0 & 0 & 2 & -2 & -3 & 5 \\ 0 & -2 & 0 & 25 & 5 & -35 \\ 1 & 0 & 0 & -5 & -1 & 6 \end{pmatrix}.$$

Перетворивши матрицю $A$ до неповного трикутно-діагонального вигляду $A'$ і застосувавши TSS-алгоритм, дістаємо таку кількість комбінацій -$L$(v)u + +$L$(u)v, які обчислюються під час побудови TSS (TSS складається з двох розв'язків (16,15,89,0,76,10) і (2,0,3,5,7,5)) цієї СЛОДР з матрицями $A$ і $A'$: для першої матриці обчислюється 31комбінація, а для другої – 8 комбінацій!!!

***Третє оптимізаційне перетворення*** випливає з аналізу ситуації в прикладі 2. Аналіз векторів, які входять до TSS, показує, що обчислення, пов'язані з наступним рівнянням системи, збільшують кількість ненульових координат в розв'язках TSS на одиницю. Якщо кількість ненульових координат в лінійній комбінації e := -$L$(v)u + $L$(u)v при обчисленнях для $i$-го рівняння системи перевищує величину $q$-($i$+1), де $q$ кількість невідомих системи, то будувати цю комбінацію непотрібно. В прикладі 2 при знаходженні TSS підсистеми з перших двох рівнянь не будуть обчислюватися розв'язки $e_3'$, $e_4'$, $e_3'$, а при обчисленні TSS всієї системи не будуть обчислюватися розв'язки $s_1'$, $s_2'$.Це прискорює обчислення TSS СЛОР. Останнє обгрунтовує така теорема.

**Теорема 3.** Нехай $S$ – СЛОДР вигляду (1) і $M_p'$ – її TSS, яка має $k$ елементів. Тоді довільний вектор $x$ із $M_p'$ такий, що $tx \geq e_i \in M_p' \setminus \{x\}$, $t \in N$ і $t \neq 0$, представляється у вигляді невід'ємної лінійної комбінації $mx = b_1e_1 + b_2e_2 + \ldots + b_{k-1}e_{k-1}$, де $m \in N$, $m \neq 0$, $b_i \in N$, i = 1, 2,…, $k$-1[8].

Обчислень комбінацій такого типу можна уникнути, якщо завчасно підрахувати кількість ненульових координат в комбінації e := -$L$(v)u + $L$(u)v. Це потребує такої модифікації підпрограми **TSSEQN1**(M,L($x$)):

```
          TSSEQN1(M,L(x))
begin
  M^0:= ∅; \ M^+:=∅; M:= ∅;
  forall e ∈ M do
    if L(e)=0 then M^0:=M^0 ∪ {e} else
      if L(e) > 0 then M^+:=M^+ ∪ {e} else M^-:=M^- ∪ {e};
      M':= M^0;
      if M^+ ≠ ∅ ∧ M^- ≠ ∅ then
        forall u ∈ M^+ do          (* де u=(u(1),…, u(q)) *)
          forall v ∈ M^- do        (* де v=(v(1),…,v(q)) *)
            z := 0;                (* підрахунок кількості ненульових координат *)
            for j:=1 to q do if (u(j) ≠ 0 ∨ v(j) ≠ 0) then z:=z+1 od
            if z ≤ (q-(i+1)) then ( e := -L(v)u + L(u)v; M':=M' ∪ {e});
  return(M');
end
```

### 3. Експериментальні результати оптимізаційних перетворень

Підводячи підсумки вищесказаному, наведемо експериментальні результати, отримані на конкретних прикладах систем.

Таблиця 1

Розмірності систем, норми коефіцієнтів та часові характеристики їх розв'язання до і після застосування оптимізаційних перетворень

| Розмірність системи | Час без оптимізації (секунди) | Час з оптимізацією (секунди) | Загальна кількість кандидатів | Кількість відкинутих кандидатів |
|---|---|---|---|---|
| 32x38 | 5.45 | 0.45 | 56240 | 45695 (81%) |
| 12x25 | 229.29 | 184.44 | 40699436 | 6002799 (14%) |
| 12x25.1 | 237.76 | 216.54 | 44543433 | 5965471 (13%) |

Загалом перше перетворення, якщо воно застосовне, дає в середньому 15-20% прискорення обчислень, друге перетворення дає в середньому 35-40% і третє перетворення – 20-25%.

**Висновки.** В роботі представлено ефективність оптимізаційних перетворень алгоритму побудови мінімальної генеруючої множини розв'язків систем лінійних однорідних рівнянь у множині натуральних чисел.

## Література

[1] *Gordan P.* Ueber die Auflösung linearen Gleidungen mit reallen Coefficienten. Mathematische Annalen, 1873, № 6, P. 23–28.
[2] *Hilbert D.* Ueber die Theorie der algebraischen Formen. Mathematische Annalen, 1890, № 36, – P. 473–534.
[3] *Clausen M., Fortenbacher A.* Efficient solution of linear diophantine equations. Journ. Symbolic Computation, 1989, v.8. № 1,2. – P. 201–216.
[4] Cont*enjean E., Devie H.* An efficient algorithm for solving systems of Diophantine equations, Inform. Comput, 1994, № 113, v. 1. – P. 143–172.
[5] *Domenjoud E.* Outils pour la deduction automatique dans les theories associatives-commutatives. Thesis de Doctorat d'Universite: Universite de Nancy I, 1991.
[6] *Pottier L.* Minimal solution of linear diophantine systems: bounds and algorithms. In Proceed. of the 4-th Intern. Conf. on Rewriting Techniques and Applications, Como, Italy, 1991, – P.162–173.
[7] *Крывый С.Л.* 0 вычислении минимального множества инвариантов сетей Петри. Ж. Искусственный интеллект, 2001, № 3. с. 199-206
[8] *Кривий С.Л.* Лінійні Діофантові обмеження та їх застосування. К: Інтерсервіс, 2021, – 260 с.
[9] *Hermann M., Juban L., Kolaitis P. G.* On the Complexity of Counting the Hilbert Basis of a Linear Diophantine System. Springer Verlag, LNCS, 1999, № 1705. – P. 13–32.

**Modification of algorithm for constructing minimal supported set of solutions of SLE over the set of natural numbers**

*Sergii Kryvyi, Oleksii Chugaenko*

*The work presents optimization transformations of the algorithm for constructing the minimum supported set of solutions of the system of linear equations in the set of natural numbers, which improve the time characteristics of this algorithm.*

Отримано 13.03.23