УДК 681.3:519.15

В. Головко
Національний університет “Львівська політехніка”
кафедра автоматизованих систем управління

МЕТОД ЗНАХОДЖЕННЯ ОПТИМАЛЬНИХ КОМБІНАТОРНИХ КОНФІГУРАЦІЙ У ВИГЛЯДІ РОЗГАЛУЖЕНИХ В’ЯЗАНОК

Розглянуто метод знаходження комбінаторних конфігурацій розгалужених в’язанок, який дає змогу перебирати всі можливі варіанти оптимального розподілу чисел для переліку множини чисел натурального ряду.

Method for finding combinatorial configurations as branched bundles, which allows enumeration of all possible arrangements of the bundles from the given set of natural numbers are regarded in this paper.

Основні властивості комбінаторних конфігурацій з розгалуженою структурою

Елементи та внутрішні зв’язки комбінаторних моделей у загальному випадку можуть утворювати порівняно з лінійкою чи кільцем складнішу структуру. В зв’язку з цим виникає питання побудови ідеальних послідовностей чисел, що утворюють будь-яку задану структуру, яка дає змогу генерувати ряд натуральних чисел у вигляді суми поруч розміщених елементів. Зупинимося коротко на структурі типу розгалуженої лінійки.

Під розгалуженою лінійкою розуміємо послідовність чисел, в якій один і той самий зв'язок між елементами є спільним більш ніж для двох елементів і не має замкнутих ланок. Ідеально розгалуженою лінійкою *n*-го порядку кратності *R* називається утворена на множині $K_n = \{k_i\}, i = \overline{1,n}$ цілих чисел розгалужена лінійка чисел, на якій всі можливі суми, включаючи значення її окремих елементів $k_1, k_2, ..., k_i, ..., k_n$, набувають значення чисел натурального ряду $1,2,...,S_l$, кожне з яких є значенням *R* різних сум з числовими кодами (p_j, q_j), що відрізняються між собою, де S_l – максимальна сума на *l*- послідовності чисел, яка належить цій розгалуженій лінійці [1].

Відомо, що існує співвідношення між параметрами ідеальної розгалуженої лінійки. Найбільше числове значення суми $S\max$ на розгалуженій лінійці при *R*=1

$$S_{\max} = S_l = S_n - S_k, \tag{1}$$

де S_l – сума всіх чисел *l*- послідовності; S_n – сума всіх чисел розгалуженої лінійки; S_k – сума всіх чисел, що не входять до складу *l*- послідовності.

Максимально можлива кількість *K* способів реалізації сум на розгалуженій лінійці визначається

$$S_n = \frac{n(n+1)}{2} \tag{2},$$

і розподіляється рівномірно для реалізації кожного з чисел натурального ряду $1,2,...,S_{\max}$ точно за *R* способами. Звідси випливає залежність

$$K = S_{\max} \cdot R. \tag{3}$$

З усіх наведених вище (1), (2), (3) формул одержуємо

$$S_n = n(n+1)/2R + S_k, \tag{4}$$

$$S_{\max} = n(n+1)/2R; \tag{5}$$

$$K = n(n+1)/2. \quad (6)$$

Наявність в (4) складової S_k створює деяку свободу під час вибору співвідношення між параметрами *n, R,* S_n за рахунок довільного вибору S_k. Оскільки при синтезі комбінаторних систем зазвичай ставиться задача мінімізації компонент, значення S_k доцільно починати вибирати з найменших чисел, щоб загальна сума S_n була якнайменшою [1].

Інформацію про власне оптимальні лінійки можна знайти в [1], а також в [3].

Алгоритм знаходження комбінаторних конфігурацій

Нижче подано два алгоритми, які застосовуються разом для отримання унікальних послідовностей зі заданої множини чисел шляхом розбиття на менші підмножини та перестановки чисел у цих підмножинах. Кількість таких підмножин – це кількість розміщення без повторень *A(n,k)*:

$$A(n,k) = \frac{n!}{k!(n-k)!}. \quad (7)$$

Для кожної знайденої підмножини чисел виконуємо перестановку її елементів і отримаємо унікальні числові набори. Кількість таких перестановок дорівнюватиме числу $P(k) = k!$. Отже, загальна кількість можливих комбінацій, кожна з яких була б потенційним варіантом розв'язку, за умови унікальності всіх комбінацій дорівнює

$$A(n,k) \cdot P(k) = \frac{n!}{k!(n-k)!} \cdot k! = \frac{n!}{(n-k)!}. \quad (8)$$

Отримані числові набори вносимо в запрограмовані таблиці суміжних сум, різні для кожного типу топології елементів на просторовій лінійці. У програмі потрібно буде додавати елементи цих таблиць за визначеними правилами і зіставляти їхні суми. Якщо в таблиці не існує повторень (кожна сума унікальна), то отримана числова послідовність буде розглядатися для однієї з можливих конфігурацій просторової лінійки з визначеним розміщенням елементів, звичайно, якщо сума цих елементів коливається в теоретично можливих межах $S_n \le S < S_{n+1}$.

Алгоритм генерування всіх *k*-елементних підмножин множини {1,…,n} у лексикографічному порядку (докладний опис роботи алгоритму наведено в монографії польського спеціаліста Вітольда Ліпського[2]). Даними для цього алгоритму є число *n* – потужність множини та число *k* – потужність генерованих підмножин. Результатом обчислень є послідовність всіх *k*-елементних підмножин множини {1,…,n}, впорядкована за лексикографічним порядком.

```
begin
    for i := 1 to k do A[i] := i; (*перша підмножина*)
    p := k;
    while p ≥ 1 do
    begin write(A[1],…, A[k]);
            if A[k] = n then p := p – 1 else p := k;
            if p ≥1 then
            for i := k downto p do A[i] := A[p]+i-p+1
    end
end
```

Алгоритм для генерування всіх послідовностей в антилексикографічному порядку (взято з [2]) наведено нижче з використанням псевдокоду. Даними для цього алгоритму є число n – потужність множини. Результат являє собою послідовність перестановок множини {1,…,n} в антилексикографічному порядку. Примітка: операція ":=:" передбачає обмін значеннями операндів.

```
1: procedure REVERSE(m);
2: begin i := 1; j := m;
3:       while i < j do
```

```
4:              begin P[i] :=: P[j]; i := i + 1; j := j – 1;
5:              end
6:  end; (*REVERSE *)
7:  procedure ANTYLEX(m);
8:  begin
9:      if m = 1 then
10:             write(P[1],…, P[n])
11:     else
12:             for i := 1 to m do
13:                     begin ANTYLEX(m – 1);
14:                             if i < m then
15:                                     begin P[i] :=: P[m]; REVERSE(m – 1)
16:                                     end
17:                     end
18: end; (*ANTYLEX*)
19: begin(*головна програма*)
20:     (*масив P – глобальний*)
21:     for i := 1 to n do P[i] := i;
22:     ANTYLEX(n)
23: end
```

Таблиця 1

Робота алгоритму за різної кількості чисел розгалуженої лінійки на заданій множині поділок {1,2,3,4,5,6,7,8,9,10,11,12,13,14}

n, од	4	5	6	7
$t,10^{-3}c$	32	156	1860	36875

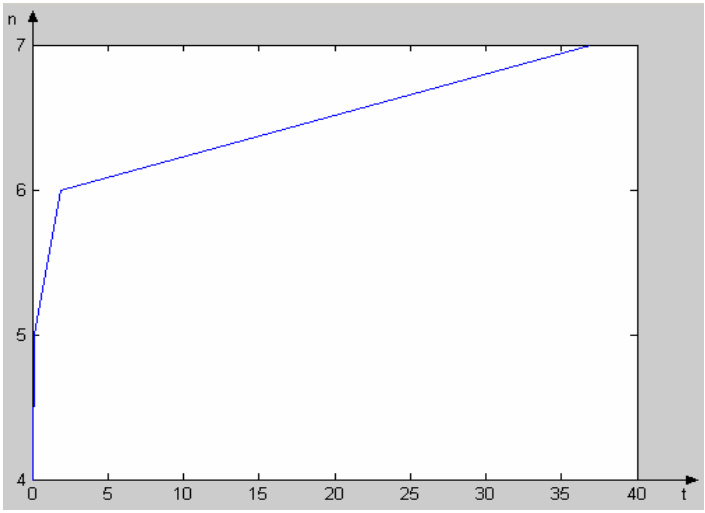

Рис. 1. Залежність часу обчислень від кількості чисел розгалуженої лінійки на заданій множині поділок {1,2,3,4,5,6,7,8,9,10,11,12,13,14}

Таблиця 2

Робота алгоритму за різної кількості чисел розгалуженої лінійки на заданій множині поділок {1,2,3,4,5,6,7,8,9,10,11,12,13}

n, од	4	5	6	7
$t,10^{-3}c$	16	93	984	7094

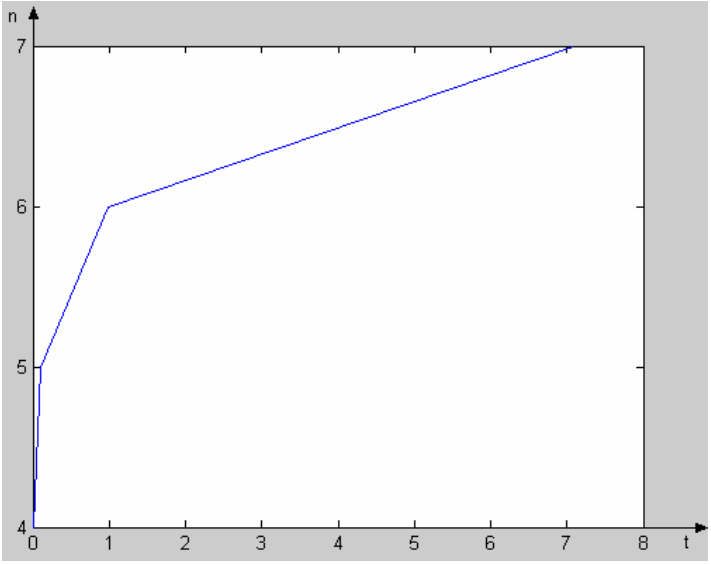

Рис. 2. Залежність часу обчислень від кількості чисел розгалуженої лінійки на заданій множині поділок {1,2,3,4,5,6,7,8,9,10,11,12,13}

Таблиця 3

Робота алгоритму за різної кількості чисел розгалуженої лінійки на заданій множині поділок {1,2,3,4,5,6,7,8,9,10,11,12}

n, од	4	5	6	7	8
$t,10^{-3}c$	15	62	547	3234	21687

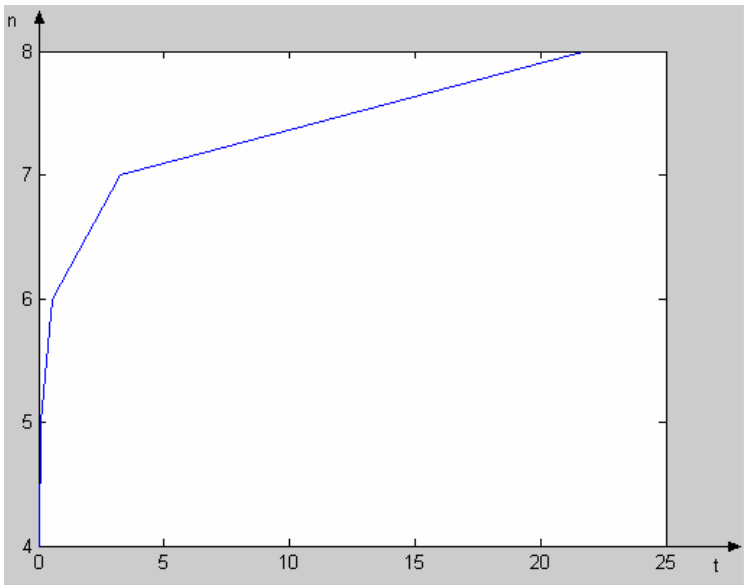

Рис.3. Залежність часу обчислень від кількості чисел розгалуженої лінійки на заданій множині поділок {1,2,3,4,5,6,7,8,9,10,11,12}

Часові затрати аналізували шляхом виконання алгоритму для генерування всіх послідовностей на трьох різних можливих множинах. Результати задокументовані в табл. 1, 2 і 3. Відповідні графіки до перерахованих таблиць показані на рис. 1, 2, 3.

Результати застосування запропонованого алгоритму

Слід зазначити, що з десяти можливих топологій на рис. 4 показано тільки чотири, для яких існує розв'язок за критерієм $S_n \le S < S_{n+1}$ і тільки для шести елементів, хоча це не є тривіальний випадок.

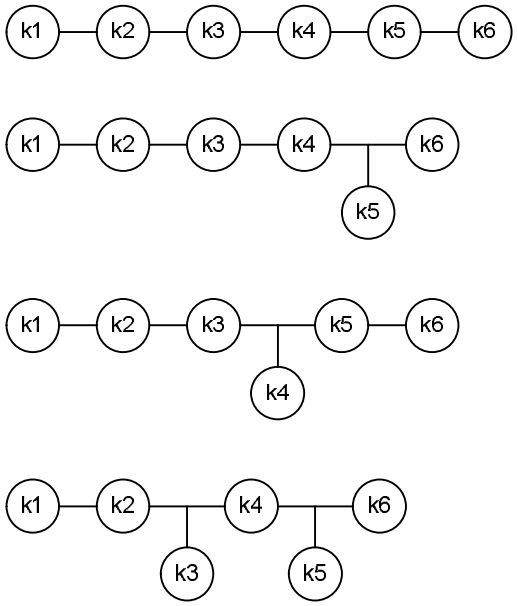

Рис. 4. Топології розгалужених лінійок, для яких існують конфігурації, $S_n < 28$

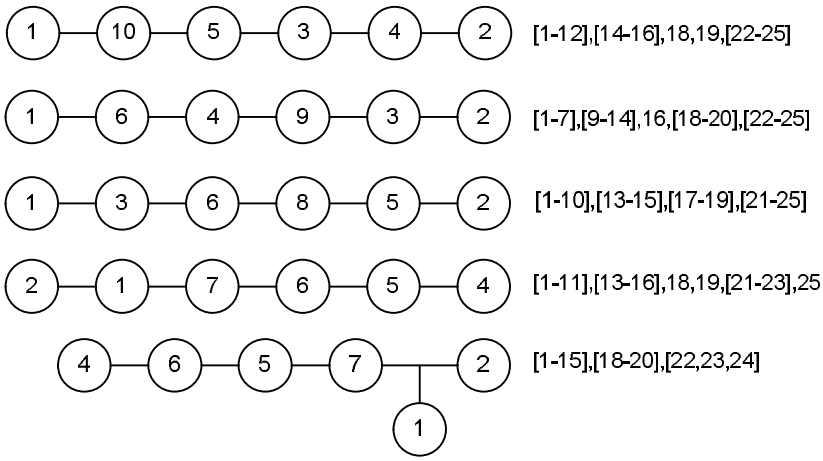

Рис. 5. Конфігурації деяких неповних розгалужених лінійок, $S_n = 25$

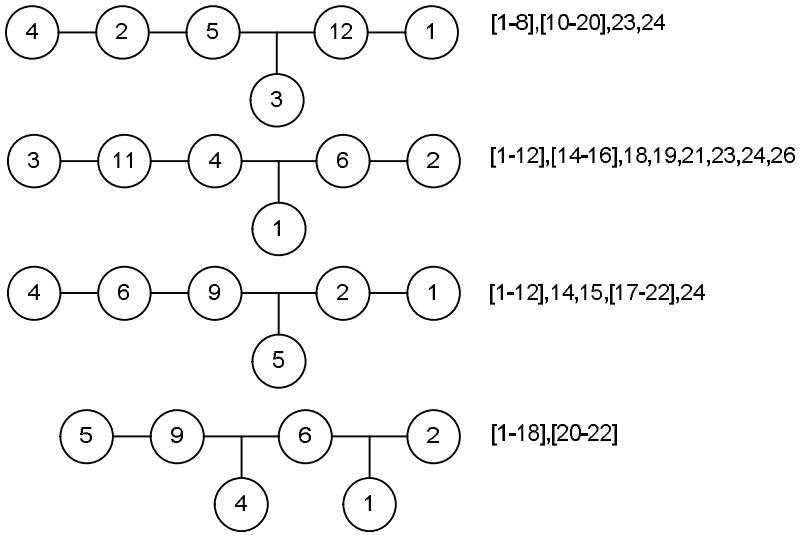

Рис.6. Конфігурації деяких неповних розгалужених лінійок, $S_n = 27$

Для генерування вхідної вибірки було взято множину натуральних чисел {1, 2, 3, 4, 5, 6, 7, 8, 9}. Загальна кількість можливих комбінацій у такому випадку становить $\frac{9!}{3!} = 60480.$ Після цього потрібно відняти всі комбінації, сума елементів яких більша за 27. Отже, остаточна кількість комбінацій, які були згенеровані, – 16560. Результати показані на рис. 5 та 6.

Отже, на розглянутих прикладах було показано оптимальні комбінаторні конфігурації на розгалужених лінійках, що існують за умови $S_n > 24$.

Висновки

Розглянуто метод знаходження оптимальних комбінаторних конфігурацій у вигляді розгалужених лінійок та два комбінаторні алгоритми, які забезпечують генерацію всіх варіантів числових наборів у вигляді доданків, складених із послідовно розташованих на в'язанці числових елементів зі заданої множини натуральних чисел, які не повторюються. На основі цих алгоритмів досліджено декілька топологій лінійки шостого порядку, які задовольняють оптимізаційний критерій за сумою числових елементів лінійки *S*,де $S_n \le S < S_{n+1}$.

Результати досліджень можна використати для синтезу електричних кіл або для відтворення електричних напруг на резистивних елементах.

1. Різник В.В. Синтез оптимальних комбінаторних схем. – Львів: Вища школа, 1989. – С. 14. 2. Липский В. Комбинаторика для программистов. – М.: Мир, 1988. – С. 21–22; С. 40–41. 3. Golomb ruler, http://en.wikipedia.org/wiki/Golomb_ruler.