ОПТИМІЗАЦІЯ РОБОТИ З ЕЛЕМЕНТАМИ ГРУПИ ПІДСТАНОВОК В ОБЧИСЛЮВАЛЬНИХ СИСТЕМАХ

М.О.Нямещук

Ужгородський державний інститут інформатики, економіки і права,
вул. Заньковецької, 89б,. Ужгород,
e-mail: gamer@ukr.net

Розглянуто питання оптимізації роботи з елементами групи підстановок в обчислювальних системах. Обговорюється аспект швидкодії при алгебраїчних операціях з підстановками, а також стиснення даних (підстановок) при збереженні в пам'яті і при передачі в мережі.

При дослідженні фізичних процесів інколи потрібно працювати зі скінченними впорядкуваннями. В [1] показано зв'язок між скінченними впорядкуваннями та елементами групи підстановок. Крім того, виникає необхідність в застосуванні теорії груп. Згідно з теоремою Келі, всяка скінченна група ізоморфна деякій групі підстановок. Тоді є доцільною побудова швидких алгоритмів для роботи з елементами групи підстановок. Також варто приділити увагу до питання стиснення даних при збереженні в постійній пам'яті або при передачі даних в мережі.

Нехай задана група підстановок S_n. Порядок групи S_n дорівнює $n!$

Як відомо, довільна підстановка $\pi \neq e$ в S_n є добутком незалежних циклів довжини ≥ 2. Цей розклад в добуток визначено однозначно з точністю до порядку слідування циклів.

Наприклад, підстановку

$$\pi = \begin{pmatrix} 12345678 \\ 24531768 \end{pmatrix}$$

ми запишемо у вигляді

$$\pi=(12435)(67)(8)=(12435)(67). \quad (1)$$

Неважко помітити, що довжина циклу не може перевищувати n, а кількість циклів – $n/2$.

Надалі будемо виконувати множення зліва направо.

Покажемо, що можна побудувати розклад підстановки в добуток циклів, елементи кожного з яких будуть слідувати в порядку зростання.

Розглянемо довільний цикл

$$a=(a_1,a_2,\ldots,a_k,a_{(k+1)},\ldots,a_m).$$

В цьому циклі існує мінімальний елемент a_t $(1\leq t\leq m)$. Цикл a буде еквівалентний циклу

$$b=(b_1,b_2,\ldots,b_k,b_{(k+1)},\ldots,b_m), \text{ де } b_1=a_t,\ b_1<b_i,$$

$$b_i\in\{a_1,a_2,\ldots,a_m\},\ i=2,3,\ldots,m.$$

Нехай

$$b_1<b_2<\ldots<b_k,\ b_{(k+1)}<b_k.$$

Тоді цикл b можна розкласти в добуток циклів:

$$(b_1,b_2,\ldots,b_k)(b_1,b_{(k+1)},b_{(k+2)},\ldots,b_m).$$

Розклавши за таким же правилом цикл

$$(b_1,b_{(k+1)},b_{(k+2)},\ldots,b_m),$$

отримаємо, що цикл b однозначно розкладається в добуток циклів, кожен з яких починається з b_1, а елементи в них розміщені в порядку зростання. Однозначність розкладу випливає з однозначності алгоритму побудови добутку.

Означення 1. Отримані цикли назвемо зростаючими.

Означення 2. Добуток циклів, у яких однаковий перший елемент назвемо великим циклом.

Твердження 1. Великі цикли – незалежні, тобто множини елементів, що містяться в різних великих циклах, не перетинаються.

Доведення. Дійсно, результат множення множників великого циклу утворює цикл, типу (1). А такі цикли – незалежні.

В [2] показано алгоритм побудови групи S_n на основі побудованої групи S_{n-1}.

Подання підстановок у побітному виді. Нехай маємо групу S_n, елементи якої подані у вигляді добутків зростаючих циклів.

Означення 3. Назвемо підстановочним байтом вектор A довжини n, елементи якого належать множині $\{1,0\}$, тобто $A=[a_1a_2\ldots a_n]$, де $a_i\in\{0,1\}$, $i=1,\ldots,n$.

Розглянемо зростаючий цикл $(b_1,b_2,\ldots,b_m)$. $b_i\in\{1,2,\ldots,n\}$. Кожному зростаючому циклу поставимо у відповідність підстановочний байт за правилом: елементи з номерами $b_1,b_2,\ldots,b_m$ дорівнюють 1, а інші – 0. Побудована відповідність є взаємно однозначною.

Означення 4. Вказане подання зростаючого циклу назвемо побітним.

Побітне подання зростаючого циклу (2357) при $n=8$, буде мати вид: [01101010].

Твердження 2. Кожному розкладу елемента групи S_n в добуток r зростаючих циклів можна однозначно співставити послідовність з r підстановочних байтів.

Розглянемо далі зростаючий цикл $(b_1,b_2,\ldots,b_m)$ і відповідний йому підстановочний байт A. Для того, щоб визначити куди переходить елемент b_i в межах цього зростаючого циклу, потрібно користуватися наступним правилом: знаходиться порядковий номер комірки в підстановочному байті A, більший за b_i, де записано "1". Якщо такого номера нема, то b_i переходить в b_1. Оскільки b_1 – мінімальний елемент в даному зростаючому циклі, то на місцях з номерами меншими за b_1 будуть записані "0".

Правило множення двох зростаючих циклів у побітному вигляді, інформацію про кількість зростаючих циклів в групі S_n і про парність підстановки можна знайти в [2].

Оскільки операції з бітами виконуються процесором значно швидше, то вказаний метод подання елементів групи підстановок дає змогу швидше, з економією ресурсної бази ЕОМ, виконувати алгебраїчні операції в групі підстановок, а, отже, оптимальніше розв'язувати задачі, які вимагають роботи з підстановками.

При розв'язуванні деяких задач, часто виникає необхідність в збереженні даних в постійній пам'яті. Вказаний метод подання підстановок дозволяє здійснювати стиск даних.

При розкладі підстановки в добуток незалежних циклів множини елементів, що містяться в них, не перетинаються. При розкладі у великий цикл, перетин множин елементів, що містяться в зростаючих циклах великого циклу, буде містити один елемент. Цей елемент є мінімальним елементом у всіх зростаючих циклах великого циклу. Це означає, що при поданні у вигляді підстановочних байтів, на місцях, де були "1", в наступних підстановочних байтах будуть "0" (крім першого (мінімального) елемента). Оскільки вказані "0" визначаються з попередніх зростаючих циклів, то їх можна опускати.

Приклад стиску. Розглянемо цикл (3867245), $n=8$.

1) Мінімальний елемент ставимо на перше місце. Маємо: (2453867);
2) Розкладаємо в добуток зростаючих циклів: (245)(238)(267);
3) Подаємо у виді підстановочних байтів:
   [01011000][01100001][01000110];

Підкреслимо "1" і відповідні їм "0".
[01011000][01100001][01000110];

Розглянемо 2-й і 3-й підстановочні байти: [01100001][01000110];

1) Відкинемо підкреслені "0". Отримаємо: [01011000][011001][0111];

Кількість біт зменшилась.

Крім того, група біт, що визначає мінімальний елемент великого циклу, повторюється в кожному підстановочному байті. Вказану групу теж можна опустити

в наступних підстановочних байтах. Тоді будемо мати: [01011000][1001][11].

Оскільки великі цикли незалежні, то множини координат, що містять "1" не перетинаються. Звідси слідує, що в наступних підстановочних байтах можна визначити координати, де будуть міститися "0".

Наприклад:

При n=8 розглянемо добуток незалежних циклів (517)(63824).

1) Мінімальні елементи ставимо на перші місця в кожному з циклів. Отримаємо: (175)(24638);
2) Розкладаємо в добуток зростаючих циклів: (17)(15)(246)(238);
3) Подаємо у вигляді підстановочних байтів (великі цикли відокремимо комами): [10000010][10001000], [01010100][01100001];

Підкреслимо "1" і відповідні їм "0" у великих циклах: [1000001̲0][1000̲1000], [0̲101̲01̲00][0̲110̲00̲01];

1) Відкинемо підкреслені "0". Отримаємо: [10000010][10001000], [10110][11001];
2) Опустивши "0" у кожному з великих циклів за правилом, яке було вказане вище, будемо мати: [10000010][000100], [10110][11].

Зауваження 1. Для того, щоб у послідовності даних визначити великий цикл, можна використати службову інформацію, де записано кількість підстановочних байтів у великому циклі.

Зауваження 2. Якщо спочатку записувати великі цикли, в яких елементів більше, то "0", які можна опустити, буде більше.

Вказаний метод подання підстановок можна застосувати для оптимізації обміну інформацією між обчислювальними станціями в комп'ютерній мережі.

Нехай потрібно реалізувати обмін підстановками між декількома обчислювальними станціями. Можна застосувати наступний спосіб передачі даних. На всіх станціях використовується один і той же алгоритм побудови групи підстановок на базі існуючої групи підстановок меншого порядку. В межах однієї групи підстановок можна вести нумерацію елементів-підстановок. Оскільки алгоритм єдиний, то номери однакових елементів будуть однакові. При передачі підстановки аналізується, що є більшим: подання підстановки чи її номер. Тоді вибирається варіант, що є меншим, і пересилається з додатковим інформаційним бітом. Інформаційний біт показує, що передається: підстановка чи її номер.

Нехай передача відбувається на обчислювальну станцію, де не можна застосувати алгоритм побудови підстановок, або є необхідність в стисненні даних, що передаються. Тоді можна застосувати метод стиснення підстановок, який був вказаний. При цьому методі затрачається час на стиск (і навпаки), але зменшується об'єм передачі даних.

Реалізація формування підстановок і дій над ними на апаратному рівні може забезпечити збільшення продуктивності обчислювальної системи, оскільки апаратна частина буде виконувати простіші операції.

Література

1. Ф.Г.Ващук, *Введение в проблему информационно-структурного моделирования процесса иследования и системы формирования психосоциальной сферы* (Хозрасчетный редакционно-издательский отдел Комитета по делам печати и информации Закарпатского облисполкома, Ужгород, 1995).
2. М.О.Нямещук, в: *Автоматика-2000, Міжнародна конференція з автоматичного управління, Львів, 11-15 вересня 2000 р., Праці в 7 томах, Том 7,* (Державний НДІ інформаційної інфраструктури, Львів, 2000), с.132–135.

OPTIMIZATION OF WORK WITH THE ELEMENTS OF PERMUTATION GROUP IN COMPUTING SYSTEMS

M.O.Nyameshchuk

Uzhhorod State Institute of Information Science, Economics and Law,
87 B, Zankovetskoi str., Uzhhorod,
e-mail: gamer@ukr.net

The problem of optimization of work with permutation group elements in computing system is considered. The speed at algebraic operation with permutations as well as data (permutation) compression at storage and networking is discussed.