Екзаменаційна робота
з теорії алгоритмів
ПМО-21
Кравець Ольги
Білет №5

① Поняття про алгоритмічні системи.

Нехай $A = \langle \varphi, \Pi \rangle$ - деякий алгоритм із системою правил Π. Алгоритм $A_F = \langle \varphi, \Pi_F \rangle$, де Π_F формальний опис системи Π, називається **формальним еквівалентом алгоритму A**.

Алгоритмічна система - це сносій задання алгоритмів у деякому фіксованому формалізмі F, який дає можливість для довільного наперед заданого алгоритму A задати його формальний еквівалент A_F.

Три типи алгоритмічних систем, які різняться початковою ідеєю стосовно того, що таке алгоритм:

1) у першому типі алгоритми пов'язують з обчисленнями і числовими функціями (рекурсивні функції)

2) у другому типі алгоритми пов'язують з детермінованим пристроєм, здатним виконувати в кожний окремий момент лише дуже примітивні операції (машина Тьюрінга)

3) у третьому типі алгоритми пов'язують з

перетворення слів у довільних алфавітах (нормальні алгоритми Маркова та канонічна система Поста)

Один із фундаментальних результатів теорії алгоритмів: усі алгоритмічні системи є рівносильними в тому сенсі, що в кожній з них описують один і той самий клас алгоритмів.

Кожна алгоритмічна система охоплює об'єкти двох категорій: - елементарні оператори
- елементарні розпізнавачі

Елементарні оператори – це прості алфавітні оператори. Якщо послідовно виконувати такі оператори, то відбудеться реалізація довільних алгоритмів у заданій алгоритмічній системі.

Елементарні розпізнавачі – використовуються для розпізнавання деяких властивостей інформації, перетворюваної алгоритмами і для зміни послідовності виконання елементарних операторів.

Для задання допустимих у системі елементарних операторів і розпізнавачів кожна система використовує свої засоби.

② Метод „Поділяй і володарюй"

Метод „Поділяй і володарюй" у поєднанні з рекурсією дає дуже часто елегантні та ефективні алгоритми. Так як сама по собі рекурсія не приводить до ефективного алгоритму.

Алгоритми вигляду „поділяй і володарюй" мають рекурсивну структуру: для розв'язування задачі вони рекурсивно викликають самі себе один або декілька разів, щоб розв'язати допоміжну задачу, яка безпосередньо стосується сформульованої.

У методі ПВ на кожному рівні рекурсії виділяють три етапи:

1. Поділ задачі на декілька підзадач.
2. Рекурсивне розв'язування цих підзадач. Коли обсяг підзадачі достатньо малий, виділені підзадачі розв'язують безпосередньо.
3. Комбінування розв'язку вихідної задачі з розв'язків допоміжних задач.

Стандартний алгоритм вигляду ПВ:

```
ПВ (дані, N, розв'язок)
    якщо (N ≤ ГраничнийРозмір)
        тоді ПрямийРозв'язок (дані, N, розв.)
        в іншому випадку
            ПідДаних (дані, N, підмнож., розмірПідмнож, M)
        для i = 1 до M
```

повторювати ПВ(підмножс.[i], розмірПідмножс[i], підрозв'язки[i])
Комбінаціярозв'язків (підрозв'язки, M, розв'язок)

ПВ для початку перевіряє, чи дуже малий розмір задачі, тобто чи більший ніж граничний розмір для того, щоб розв. ми найш. ПрямийРозв.

В разі того, що задача дуже велика, ми для початку викликаємо ПодіДаних (ця процедура розб'є вхідні дані на декілька малих наборів підмножин). Набори можуть бути різні або однакові в розмірі. Після цього викликаємо рекурсивний виклик ПВ на кожному з наших вхідних наборів, процедура Комб. Розв. зводить отримані дані в один.

Формула визначення складності алгоритмів ПВ:

$$T_{ПВ}(N) = \begin{cases} T_{ПР}(N), \text{ при } N < \text{Гранич. Розмір} \\ T_{ПД}(N) + \sum_{i=1}^{M} T_{ПВ}(\text{розмПідмножс.}[i]) + T_{кр}(N) \text{ в.ін.в} \end{cases}$$

$T_{ПВ}$ - складність ПВ
$T_{ПР}$ - складність алгоритму Прямий розв.
$T_{ПД}$ - склад. ал. Подсл Даних
$T_{кр}$ - склад. КомбінаціяРозв.

③ Створити машину Тьюрінга для обчислення функції $I_4^5(x_1, x_2, x_3, x_4, x_5) = x_4$, якщо x_1, x_2, x_3, x_4, x_5 — числа записані в алфавіті $A = \{1\}$, а між числами x_1, x_2, x_3, x_4, x_5 ставиться *.

	1	*	λ
q_0	$1q_0R$	$1q_1R$	λq_0R
q_1	$1q_1R$	$1q_2R$	–
q_2	$1q_2R$	$1q_3R$	–
q_3	$1q_3R$	$1q_4R$	–
q_4	$1q_4R$	–	q_F

Приклад: 1*1*1*1*1

1*1*1*1*1

*1*1*1*1

1*1*1*1

*1*1*1

1*1*1

*1*1

1*1

1*1

1λ1

1