Київський національний університет імені Тараса Шевченка

Факультет комп`ютерних наук та кібернетики

Кафедра інтелектуальних програмних систем

Математичні основи захисту інформації

Лабораторна робота №1

Варіант №1

Виконали студенти 4-го курсу

Групи ІПС-42

Пащенко Дмитро Вікторович

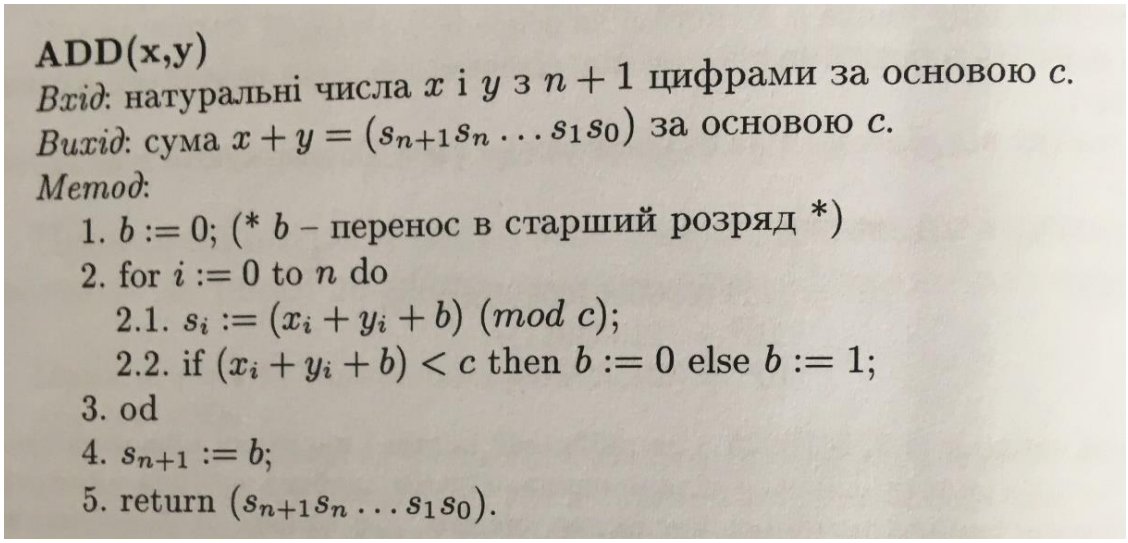
Бондарець Дарина Володимирівна

Київ - 2022

Завдання

Алгоритм ADD(x,y) – додавання натуральних чисел x та y.

Розібратись, як працює алгоритм, заданий таким псевдокодом:



ADD(x,y)
Вхід: натуральні числа x і y з $n+1$ цифрами за основою c.
Вихід: сума $x + y = (s_{n+1}s_n \dots s_1 s_0)$ за основою c.
Метод:
1. $b := 0$; (* b – перенос в старший розряд *)
2. for $i := 0$ to n do
   2.1. $s_i := (x_i + y_i + b)\ (mod\ c)$;
   2.2. if $(x_i + y_i + b) < c$ then $b := 0$ else $b := 1$;
3. od
4. $s_{n+1} := b$;
5. return $(s_{n+1}s_n \dots s_1 s_0)$.

- Оцінити арифметичну складність цього алгоритму.
- Написати програму для 3-стрічкової детермінованої машини Тьюрінга (ДМТ).
- Оцінити складність програми ДМТ.
- Виконану роботу описати у звіті.

Теорія

ДМТ із к стрічками (k >= 1) називається четвірка M — (K, X, δ, s0), де K і X ті ж, що і в означенні звичайної однострічкової ДМТ, а функція переходів δ, яка називається програмою, визначає наступний стан таким чином:

$\delta : K \times X_k \to K \times (X \times \{l, r, t\})k$

де $\delta(s, y_1, \dots, y_k) = (s', z_1, d_1, \dots, z_k, d_k)$ означає, що коли ДМТ в деякий момент перебуває в стані s, головка на першій стрічці оглядає символ y_1 і т. д., головка на k-й стрічці оглядає символ y_k, то в наступний момент ДМТ перебуватиме в стані s', головка на першій стрічці запише символ z_1 замість символа y_1 і

перейде або залишиться на місці залежно від значення d_1 і т. д., головка на k-й стрічці запише символ z_k замість символа y_k і перейде або залишиться в тій самій позиції, залежно від значення d_k.

Результат роботи багатострічкової ДМТ визначається так само, як і для звичайної ДМТ, з тією лише відмінністю, що результат обчислень словарної функції після зупинки ДМТ записується на останній k-й стрічці.

Алгоритм ADD(x,y) – додавання натуральних чисел x та y.

Розберемо, як працює алгоритм, заданий таким псевдокодом:

ADD(x,y)
Вхід: натуральні числа x і y з $n+1$ цифрами за основою c.
Вихід: сума $x + y = (s_{n+1}s_n \dots s_1s_0)$ за основою c.
Метод:
1. $b := 0$; (* b – перенос в старший розряд *)
2. for $i := 0$ to n do
   2.1. $s_i := (x_i + y_i + b) \ (mod\ c)$;
   2.2. if $(x_i + y_i + b) < c$ then $b := 0$ else $b := 1$;
3. od
4. $s_{n+1} := b$;
5. return $(s_{n+1}s_n \dots s_1s_0)$.

Нехай числа **x** та **y** представлені у вигляді $(x_n x_{n-1} \dots x_0)$ та $(y_n y_{n-1} \dots y_0)$ відповідно. Нехай результат алгоритму представлені у вигляді $(s_{n+1}s_n \dots s_1s_0)$ за основою c. Тоді виконуємо наступні дії:

Дія 1:

На вхід отримуємо змінні **x, y** за основою **c**, та задаємо змінну **b=0**, яка вказує на перенос в старший розряд.

Дія 2:

Проходимся по кожному і-тому розряду чисел **x** і **y** (де $i = \overline{0..n}$) та виконуємо наступні дії:

Дія 2.1:

В цьому ж тілі циклу обраховуємо і-й розряд за формулою

$s_i = x_i + y_i + b \ (mod\ c)$

Дія 2.2:

В цьому ж тілі циклу перевіряємо чи $(x_i + y_i + b) < c$. Якщо умова виконується, тоді змінній b передаємо значення 0, інакше b = 1.

Дія 4: s_{n+1} присвоюємо значення b (перенос в старший розряд).

Дія 5: В результаті отримаємо суму чисел **x** та **y** представлену у вигляді $(s_{n+1}s_n \dots s_1s_0)$.

Оцінка арифметичної складності алгоритму

Часова складність даного алгоритму додавання двох натуральних чисел складає O(n). Складність по пам'яті складатиме O(n).

Опис програми для для 3-стрічкової детермінованої машини Тьюрінга

Початок програми.

$$(q_0, \Delta, \Delta, \Delta) \to (q_{copy}, \Delta, r, \Delta, r, \Delta, r)$$

Копіюємо x на другу стрічку.

$$(q_{copy}, 0, \#, \#) \to (q_{copy}, 0, r, 0, r, \#, t)$$

$$(q_{copy}, 1, \#, \#) \to (q_{copy}, 1, r, 1, r, \#, t)$$

...

$$(q_{copy}, c-1, \#, \#) \to (q_{copy}, c-1, r, c-1, r, \#, t)$$

$$(q_{copy}, \#, \#, \#) \to (q_{shift}, \#, r, \#, t, \#, t)$$

Зміщуємося в кінець y.

$$(q_{shift}, 0, \#, \#) \to (q_{shift}, 0, r, \#, t, \#, t)$$

...

$$(q_{shift}, c-1, \#, \#) \to (q_{shift}, c-1, r, \#, t, \#, t)$$

$$(q_{shift}, \#, \#, \#) \to (q_{check\_digit}, \#, l, 0, t, \#, t)$$

Описуємо процедуру додавання.

Виконуємо перенос в наступний розряд.

$$(q_{check\_digit}, \_, 0, \#) \to (q_{shift\_right}, \_, t, 0, l, \#, t)$$

$$(q_{check\_digit}, \_, 1, \#) \to (q_{add\_digit}, \_, t, 0, l, \#, t)$$

$$(q_{shift\_right}, \_, 0, \#) \to (q_{shift\_right\_0}, \_, t, 0, r, \#, t)$$

$$(q_{shift\_right}, \_, 1, \#) \to (q_{shift\_right\_1}, \_, t, 0, r, \#, t)$$

...

$$(q_{shift\_right}, \_, c-1, \#) \to (q_{shift\_right\_<c-1>}, \_, t, 0, r, \#, t)$$

$$(q_{shift\_right}, \_, \Delta, \#) \to (q_{copy\_result\_start}, \_, t, \Delta, r, \#, t)$$

$$(q_{add\_digit}, \_, 0, \#) \to (q_{shift\_right\_1}, \_, t, 0, r, \#, t)$$

$$(q_{add\_digit}, \_, 1, \#) \to (q_{shift\_right\_2}, \_, t, 0, r, \#, t)$$

...

$$(q_{add\_digit}, \_, c-2, \#) \to (q_{shift\_right\_<c-1>}, \_, t, 0, r, \#, t)$$

$$(q_{add\_digit}, \_, c-1, \#) \to (q_{shift\_right\_0}, \_, t, 1, r, \#, t)$$

$$(q_{add\_digit}, \_, \Delta, \#) \to (q_{return\_digit}, \_, t, \Delta, r, \#, t)$$

$$(q_{return\_digit}, \#, 0, \#) \to (q_{copy\_result}, \#, t, 1, t, \#, t)$$

$$(q_{shift\_right\_0}, \_, 0, \#) \to (q_{add}, \_, t, 0, t, \#, t)$$

$$(q_{shift\_right\_1}, \_, 0, \#) \to (q_{add}, \_, t, 1, t, \#, t)$$

...

$$(q_{shift\_right\_<c-1>}, \_, 0, \#) \to (q_{add}, \_, t, c-1, t, \#, t)$$

Виконуємо додавання x_i та y_i.

$$(q_{add}, 0,0, \#) \to (q_{check\_digit}, 0, l, 0, l, \#, t)$$

$$(q_{add}, 0,1, \#) \to (q_{check\_digit}, 0, l, 1, l, \#, t)$$

$$(q_{add}, 1,0, \#) \to (q_{check\_digit}, 1, l, 1, l, \#, t)$$

...

$$\begin{cases} (q_{add}, a_k, b_k, \#) \to (q_{carry\_digit}, a_k, l, (a_k + b_k)\text{mod(c)}, l, \#, t), a_k + b_k \geq c \\ (q_{add}, a_k, b_k, \#) \to (q_{check\_digit}, a_k, l, a_k + b_k, l, \#, t), a_k + b_k < c \end{cases}$$

$$\begin{cases} (q_{add}, a_{k+1}, b_{k+1}, \#) \to (q_{carry\_digit}, a_{k+1}, l, (a_{k+1} + b_{k+1})\text{mod(c)}, l, \#, t), a_{k+1} + b_{k+1} \geq c \\ (q_{add}, a_{k+1}, b_{k+1}, \#) \to (q_{check\_digit}, a_{k+1}, l, a_{k+1} + b_{k+1}, l, \#, t), a_{k+1} + b_{k+1} < c \end{cases}$$

Причому: $a_k + b_k < a_{k+1} + b_{k+1} \vee a_k < a_{k+1}$ (нумерація Кантора).

...

$$(q_{add}, c-1, c-1, \#) \to (q_{carry\_digit}, c-1, l, c-2, l, \#, t)$$

$$(q_{carry\_digit}, \_, 0, \#) \to (q_{check\_digit}, \_, l, 1, t, \#, t)$$

Копіюємо результат на третю стрічку.

$$(q_{copy\_result\_start}, \#, 0, \#) \to (q_{copy\_result}, \#, t, 0, r, \#, t)$$

$$(q_{copy\_result}, \#, 0, \#) \to (q_{copy\_result}, \#, t, 0, r, 0, r)$$

$$(q_{copy\_result}, \#, 1, \#) \rightarrow (q_{copy\_result}, \#, t, 1, r, 1, r)$$

...

$$(q_{copy\_result}, \#, c-1, \#) \rightarrow (q_{copy\_result}, \#, t, c-1, r, c-1, r)$$

$$(q_{copy\_result}, \#, \#, \#) \rightarrow (q_{end}, \#, t, \#, t, \#, t)$$

Приклад для c=2 (двійкове числення)

Програма ДМТ

$$(q_0, \Delta, \Delta, \Delta) \rightarrow (q_{copy}, \Delta, r, \Delta, r, \Delta, r)$$

$$(q_{copy}, 0, \#, \#) \rightarrow (q_{copy}, 0, r, 0, r, \#, t)$$

$$(q_{copy}, 1, \#, \#) \rightarrow (q_{copy}, 1, r, 1, r, \#, t)$$

$$(q_{copy}, \#, \#, \#) \rightarrow (q_{shift}, \#, r, \#, t, \#, t)$$

$$(q_{shift}, 0, \#, \#) \rightarrow (q_{shift}, 0, r, \#, t, \#, t)$$

$$(q_{shift}, 1, \#, \#) \rightarrow (q_{shift}, 1, r, \#, t, \#, t)$$

$$(q_{shift}, \#, \#, \#) \rightarrow (q_{check\_digit}, \#, l, 0, t, \#, t)$$

$$(q_{check\_digit}, \_, 0, \#) \rightarrow (q_{shift\_right}, \_, t, 0, l, \#, t)$$

$$(q_{check\_digit}, \_, 1, \#) \rightarrow (q_{add\_digit}, \_, t, 0, l, \#, t)$$

$$(q_{shift\_right}, \_, 0, \#) \rightarrow (q_{shift\_right\_0}, \_, t, 0, r, \#, t)$$

$$(q_{shift\_right}, \_, 1, \#) \rightarrow (q_{shift\_right\_1}, \_, t, 0, r, \#, t)$$

$$(q_{shift\_right}, \_, \Delta, \#) \rightarrow (q_{copy\_result\_start}, \_, t, \Delta, r, \#, t)$$

$$(q_{add\_digit}, \_, 0, \#) \rightarrow (q_{shift\_right\_1}, \_, t, 0, r, \#, t)$$

$$(q_{add\_digit}, \_, 1, \#) \rightarrow (q_{shift\_right\_0}, \_, t, 1, r, \#, t)$$

$$(q_{add\_digit}, \_, \Delta, \#) \rightarrow (q_{return\_digit}, \_, t, \Delta, r, \#, t)$$

$$(q_{shift\_right\_0}, \_, 0, \#) \rightarrow (q_{add}, \_, t, 0, t, \#, t)$$

$$(q_{shift\_right\_1}, \_, 0, \#) \rightarrow (q_{add}, \_, t, 1, t, \#, t)$$

$$(q_{add}, 0, 0, \#) \rightarrow (q_{check\_digit}, 0, l, 0, l, \#, t)$$

$$(q_{add}, 0, 1, \#) \rightarrow (q_{check\_digit}, 0, l, 1, l, \#, t)$$

$$(q_{add}, 1, 0, \#) \rightarrow (q_{check\_digit}, 1, l, 1, l, \#, t)$$

$$(q_{add}, 1, 1, \#) \rightarrow (q_{carry\_digit}, 1, l, 0, l, \#, t)$$

$$(q_{carry\_digit}, \_, 0, \#) \rightarrow (q_{check\_digit}, \_, l, 1, t, \#, t)$$

$$(q_{copy\_result\_start}, \#, 0, \#) \rightarrow (q_{copy\_result}, \#, t, 0, r, \#, t)$$

$$(q_{copy\_result}, \#, 0, \#) \rightarrow (q_{copy\_result}, \#, t, 0, r, 0, r)$$

$$(q_{copy\_result}, \#, 1, \#) \rightarrow (q_{copy\_result}, \#, t, 1, r, 1, r)$$

$$(q_{copy\_result}, \#, \#, \#) \rightarrow (q_{end}, \#, t, \#, t, \#, t)$$

Приклад.

$$c = 2, n = 3, x = 111_2, y = 101_2$$

q_0

Δ	1	1	1	#	1	0	1	#
Δ	#	#	#	#	#	#	#	#
Δ	#	#	#	#	#	#	#	#

q_{copy}

Δ	1	1	1	#	1	0	1	#
Δ	#	#	#	#	#	#	#	#
Δ	#	#	#	#	#	#	#	#

q_{copy}

Δ	1	1	1	#	1	0	1	#
Δ	1	#	#	#	#	#	#	#
Δ	#	#	#	#	#	#	#	#

q_{copy}

Δ	1	1	1	#	1	0	1	#
Δ	1	1	#	#	#	#	#	#
Δ	#	#	#	#	#	#	#	#

q_{copy}

Δ	1	1	1	#	1	0	1	#
Δ	1	1	1	#	#	#	#	#
Δ	#	#	#	#	#	#	#	#

q_{shift}

Δ	1	1	1	#	1	0	1	#
Δ	1	1	1	#	#	#	#	#
Δ	#	#	#	#	#	#	#	#

q_{shift}

Δ	1	1	1	#	1	0	1	#
Δ	1	1	1	#	#	#	#	#
Δ	#	#	#	#	#	#	#	#

q_{shift}

Δ	1	1	1	#	1	0	1	#
Δ	1	1	1	#	#	#	#	#
Δ	#	#	#	#	#	#	#	#

q_{shift}

Δ	1	1	1	#	1	0	1	#
Δ	1	1	1	#	#	#	#	#
Δ	#	#	#	#	#	#	#	#

q_{check_digit}

Δ	1	1	1	#	1	0	1	#
Δ	1	1	1	0	#	#	#	#
Δ	#	#	#	#	#	#	#	#

q_{shift_right}

Δ	1	1	1	#	1	0	1	#
Δ	1	1	1	0	#	#	#	#
Δ	#	#	#	#	#	#	#	#

$q_{shift\_right\_1}$

Δ	1	1	1	#	1	0	1	#
Δ	1	1	0	0	#	#	#	#
Δ	#	#	#	#	#	#	#	#

q_{add}

Δ	1	1	1	#	1	0	1	#
Δ	1	1	0	1	#	#	#	#
Δ	#	#	#	#	#	#	#	#

q_{carry_digit}

Δ	1	1	1	#	1	0	1	#
Δ	1	1	0	0	#	#	#	#
Δ	#	#	#	#	#	#	#	#

q_{check_digit}

Δ	1	1	1	#	1	0	1	#
Δ	1	1	1	0	#	#	#	#
Δ	#	#	#	#	#	#	#	#

q_{add_digit}

Δ	1	1	1	#	1	0	1	#
Δ	1	1	0	0	#	#	#	#
Δ	#	#	#	#	#	#	#	#

$q_{shift\_right\_0}$

Δ	1	1	1	#	1	0	1	#
Δ	1	1	0	0	#	#	#	#
Δ	#	#	#	#	#	#	#	#

q_{add}

Δ	1	1	1	#	1	0	1	#
Δ	1	1	0	0	#	#	#	#
Δ	#	#	#	#	#	#	#	#

q_{check_digit}

Δ	1	1	1	#	1	0	1	#
Δ	1	1	0	0	#	#	#	#
Δ	#	#	#	#	#	#	#	#

q_{add_digit}

Δ	1	1	1	#	1	0	1	#
Δ	1	0	0	0	#	#	#	#
Δ	#	#	#	#	#	#	#	#

$q_{shift\_right\_0}$

Δ	1	1	1	#	1	0	1	#
Δ	1	0	0	0	#	#	#	#
Δ	#	#	#	#	#	#	#	#

q_{add}

Δ	1	1	1	#	1	0	1	#
Δ	1	0	0	0	#	#	#	#
Δ	#	#	#	#	#	#	#	#

q_{check_digit}

Δ	1	1	1	#	1	0	1	#
Δ	1	1	0	0	#	#	#	#
Δ	#	#	#	#	#	#	#	#

q_{add_digit}

Δ	1	1	1	#	1	0	1	#
Δ	0	1	0	0	#	#	#	#
Δ	#	#	#	#	#	#	#	#

q_{return_digit}

Δ	1	1	1	#	1	0	1	#
Δ	0	1	0	0	#	#	#	#
Δ	#	#	#	#	#	#	#	#

q_{copy_result}

Δ	1	1	1	#	1	0	1	#
Δ	1	1	0	0	#	#	#	#
Δ	#	#	#	#	#	#	#	#

q_{copy_result}

Δ	1	1	1	#	1	0	1	#
Δ	1	1	0	0	#	#	#	#
Δ	1	#	#	#	#	#	#	#

q_{copy_result}

Δ	1	1	1	#	1	0	1	#
Δ	1	1	0	0	#	#	#	#
Δ	1	1	#	#	#	#	#	#

q_{copy_result}

Δ	1	1	1	#	1	0	1	#
Δ	1	1	0	0	#	#	#	#
Δ	1	1	0	#	#	#	#	#

q_{copy_result}

Δ	1	1	1	#	1	0	1	#
Δ	1	1	0	0	#	#	#	#
Δ	1	1	0	0	#	#	#	#

q_{end}

Δ	1	1	1	#	1	0	1	#
Δ	1	1	0	0	#	#	#	#
Δ	1	1	0	0	#	#	#	#

Аналіз складності програми для ДМТ

На кожній ітерації додавання розрядів відбувається по 4 операції. Кількість розрядів = n. Звідси TIME(4*n)=TIME(n).

Кількість потрібних комірок робочої стрічки ДМТ обмежується (n+1), де n - кількість розрядів вхідних чисел. Звідси SPACE(n+1)=SPACE(n).

Перелік літературних джерел

- «Математичні основи захисту інформації.» С.Л. Кривий.