Міністерство освіти і науки України
Національний Авіаційний Університет
Факультет комп'ютерних наук та технологій

Кафедра прикладної математики

Звіт

Про виконання лабораторної роботи
З предмету «Мови формальних специфікацій»
На тему: Складові схеми Z

Виконав:
Студент 351 групи
Штуль В.С.
Перевірив:
Піскунов О. Г.

м. Київ
2023

Зміст

Розділ 1

Вступ

1.1 Мета завдання

Розробка схеми алгоритму єгипетського множення Ахмеса-Степанова.

1.2 Постановка завдання

• Додати до преамбули документа пакет та налаштування для коректного відображення символів мови Z (див. 2.5.1);

• Взяти за основу схеми алгоритму єгипетського множення Ахмеса-Степанова розроблені в [54, Схеми у стилі утиліти ZTC];

• Замінити у схемах тип intZ:
intZ == seq1 Char
```
forall n: intZ @ dom ( n rres {'-', '+'} ) subseteq {1}
and # (n rres digits) > 0
тип цілих чисел Z;.
```

• У алгоритмах, що записуються, використовувати операторний стиль запису замість функціонального, тобто замість sum(a, b) використовувати a + b;

• До звіту включити обидві версії специфікації: тестову версію та версію для LaTex.[1]

Розділ 2

Теоретична частина

В алгоритмі єгипетського множення використовується вже розроблене додавання, функція поділу на два (half) та функція перевірки парності (odd) (schema Functions).

• функція toN відображає символ цифри у відповідне число;

• частково певна функція toD відображає число від 0 до 9 у відповідну цифру;

• null – позначення відсутнього символу, аналогічно мові SQL. Тепер можна описати алгоритм поділу навпіл, з відкиданням дробової частини(schema Half).

У поданій схемі:

•функція half видаляє знак з числа, виконує розподіл отриманого натурального числа навпіл за допомогою функції half0 на 2 і у разі знамінус дописує його до результату.

•функція half0 виконує розподіл цифри зі старшого розряду на 2 з урахуванням залишку від розподілу старшої цифри, передаючи цифри молодшихрозрядів і залишок від розподілу рекурсивному виклику себе.

•функція `half1` виконує розподіл числа (від 0 до 9) з урахуванням залишку `a`-відділення старшого розряду на 2 `((ch+a*10)/2)`. Додатково повертає залишок від свого поділу для молодшого розряду `(i mod 2)`.

Алгоритм функції парне - непарне очевидний і цікавий тільки з точки зору використання мови Z (schema Odd).

Тепер можна записати рекурсивну версію алгоритму множення Ахмеса-Степанова з акумулюванням (Schema Mult_acc3).

Алгорим з акумулюванням добре працює на непарних числах, тому для парних чисел частину роботи можна робити без нього (Schema Multiply).

Розділ 3

Практична частина

3.1 Розробка схеми алгоритму єгипетського множення Ахмеса-Степанова

Functions

$$
\begin{array}{l}
bool : \{0, 1\} \\
digits : \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\} \\
intZ : \mathbb{Z} \\
null : Char \\
toN : digits \rightarrow \mathbb{N} \\
toD : \mathbb{N} \nrightarrow digits \\
\hline
\forall\, ch : digits \bullet toN(ch) = ascii\_of(ch) - 48 \\
\forall\, x : \mathbb{N} \bullet toD(x) = \mathbf{if}\ x \in 0\,..\,9\ \mathbf{then} \\
\qquad ascii\_char(x + 48)\ \mathbf{else}\ null
\end{array}
$$

Half

$half : intZ \rightarrow intZ$
$half0 : \text{seq}\, digits \times bool \rightarrow \text{seq}\, digits \times bool$
$half1 : digits \times bool \rightarrow digits \times bool$

$\forall\, n : intZ \bullet half(n) = \textbf{if}\ head n = minus\ \textbf{then}\ \langle minus \rangle \frown first(half0(tail n, 0))\ \textbf{else}\ (\textbf{if}\ head n = plus\ \textbf{then}\ first(half0(tail n, 0))\ \textbf{else}\ first(half0(n, 0)))$

$\forall\, n : \text{seq}\, digits;\ a : bool \bullet \textbf{let}\ r == half1(head\, n, a) \bullet half0(n, a) = \textbf{if}\ \#n > 0\ \textbf{then}\ (\langle first r \rangle \frown first(half0(tail n, second r)), 0)\ \textbf{else}\ (\langle\rangle, 0)$

$\forall\, ch : digits;\ a : bool \bullet \textbf{let}\ i == toN(ch) + a * 10 \bullet half1(ch, a) = (toD(i \operatorname{div} 2), i \bmod 2)$

Odd

$odd : intZ \rightarrow bool$

$\forall\, n : intZ \bullet odd(n) = toN(last n) \bmod 2$

Mult_acc3

$intZ : \mathbb{Z}$
$mult\_acc3 : intZ \times intZ \times intZ \rightarrow intZ$

$\forall\, r, n, a : intZ \bullet \textbf{let}\ n2 == half(diff(n, one)); a2 == a + a \bullet mult\_acc3(r, n, a) = \textbf{if}\ odd(n) = 1\ \textbf{then}\ (\textbf{if}\ (cmp(n, one) = 1)\ \textbf{then}\ r + a\ \textbf{else}\ mult\_acc3(r + a, n2, a2))\ \textbf{else}\ mult\_acc3(r, n2, a2)$

Multiply

$intZ : \mathbb{Z}$
$multiply3 : intZ \times intZ \rightarrow intZ$

$\forall\, n, a : intZ \bullet multiply3(n, a) = \textbf{if}\ (\neg\, odd(n) = 1)\ \textbf{then}\ multiply3(half(n), a + a)\ \textbf{else}\ (\textbf{if}\ (cmp(n, one) = 1)\ \textbf{then}\ a\ \textbf{else}\ mult\_acc3(a, half(diff(n, one)), a + a))$

3.2 Розробка схеми алгоритму єгипетського множення Ахмеса-Степанова у текстовому вигляді

```
\begin{spec}

\begin{schema}{Functions}
bool : \{ 0, 1 \} \\
digits : \{ 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 \} \\
intZ : \num \\
null : Char \\
toN : digits \fun \nat \\
toD : \nat \pfun digits
\where
\forall ch : digits @ toN (ch) = ascii\_of (ch) - 48 \\
\forall x : \nat @ toD (x) = \zif x \in 0 \upto 9 \zthen \\
\t1 ascii\_char (x + 48) \zelse null
\end{schema}

\begin{schema}{Half}
half : intZ \fun intZ \\
half0 : \seq digits \cross bool \fun \seq digits \cross \\
\t1 bool \\
half1 : digits \cross bool \fun digits \cross bool
\where
\forall n : intZ @ half (n) = \zif head n = minus \zthen \\
\t1 \langle minus \rangle \cat first (half0 (tail n, 0)) \\
\t1 \zelse (\zif head n = plus \zthen first (half0 (tail n, \\
\t1 0)) \zelse first (half0 (n, 0))) \\
\forall n : \seq digits; a : bool @ \zlet r==half1 (head \\
\t1 n, a) @ half0 (n, a) = \zif \# n > 0 \zthen (\langle \\
\t1 first r \rangle \cat first (half0 (tail n, second r)), \\
\t1 0) \zelse (\langle \rangle, 0) \\
\forall ch : digits; a : bool @ \zlet i==toN (ch) + a * 10 \\
\t1 @ half1 (ch, a) = (toD (i \div 2), i \mod 2)
\end{schema}

\begin{schema}{Odd}
odd : intZ \fun bool
\where
\forall n : intZ @ odd (n) = toN (last n) \mod 2
\end{schema}

\begin{schema}{Mult\_acc3}
intZ : \num \\
```

```
mult\_acc3 : intZ \cross intZ \cross intZ \fun intZ
\where
\forall r, n, a : intZ @ \zlet n2==half (diff (n, one)); \\
\t1 a2==a + a @ mult\_acc3 (r, n, a) = \zif odd (n) = 1 \\
\t1 \zthen (\zif (cmp (n, one) = 1) \zthen r + a \zelse \\
\t1 mult\_acc3 (r + a, n2, a2)) \zelse mult\_acc3 (r, n2, \\
\t1 a2)
\end{schema}

\begin{schema}{Multiply}
intZ : \num \\
multiply3 : intZ \cross intZ \fun intZ
\where
\forall n, a : intZ @ multiply3 (n, a) = \zif (\lnot odd \\
\t1 (n) = 1) \zthen multiply3 (half (n), a + a) \zelse \\
\t1 (\zif (cmp (n, one) = 1) \zthen a \zelse mult\_acc3 (a, \\
\t1 half (diff (n, one)), a + a))
\end{schema}

\end{spec}
```

[2] [1]

Перелік джерел посилання

[1] Alexei Piskunov. Документирование процесса разработки ПО, 08 2021.

[2] У.И.Мартыненко А.Г.Пискунов. Умножение с произвольной точностью вкольце целых, 02 2023.