УДК 004
© О. Овсяк[1], В. Овсяк[2], Ю. Петрушка[2], 2012

ОДНООСНОВНА АЛГЕБРА АЛГОРИТМІВ

Над логічними змінними і значеннями табличним і аксіоматичним методами дано означення одноосновної алгебри алгоритмів першого порядку. Аксіоматичним методом подана дефініція одноосновної алгебри алгоритмів другого порядку.

The attribute of the monobasis algorithms algebra of the first sequence has been made by the logical variables and valuesof the tabulation and axiomatic methods. The definition of the monobasic algorithms algebra of second sequence has been provided by axiomatic method.

1. ВСТУП І ФОРМУЛЮВАННЯ ЗАДАЧІ

Математичною основою інформаційних технологій є теорія алгоритмів. Найвідомішими і найчастіше використовуваними методами подання алгоритмів є вербальний і блок-схемний. Крім них відомі ще інші методи, які за використовуваними ними засобами можна поділити на дві групи. Це методи, які не використовують алгебричну методологію та методи, які її використовують. Історично неалгебричні методи появилися першими. До них, крім уже вище згаданих методів вербального і блок-схемного подання алгоритмів, належать ще такі, як рекурсивних функцій [1], рахунку лямбда [2], машин Т'юрінґа [3] і Поста [4], алгоритмів Маркова [5], машин Колмогорова [6], Шеґаге [7], Ахо – Ульмана – Хопкрофта [8], універсальних алгоритмів Крініцкого [9].

Відомо [10, 11], що алгебричними методами подання алгоритмів є система алгоритмічних алгебр Глушкова, алгебра алгоритміки Цейтліна (модифікована система алгоритмічних алгебр Глушкова), алгебра Дейкстри, алгебра Калужніна і алгебра Янова, а кожна із них має відповідний клон. Алгебра алгоритміки є дворівневою системою з неінтерпретованими схемами верхнього рівня і необхідністю побудови прикладних алгебр нижнього (спеціалізованого) рівня [10, 11].

Алгебричні у порівнянні з не алгебричними методами подання алгоритмів мають ту перевагу, що над формулами алгоритмів, так як над будь-якими математичними виразами, можуть бути виконані тотожні перетворення на підставі властивостей операцій. Результатами цих перетворень можуть бути зменшення затрат пам'яті на зберігання та

[1] Київський національний університет культури і мистецтв
[2] Українська академія друкарства

часу виконання, автоматизація процесів перетворень, комп'ютерного генерування програмного коду з формул алгоритмів, можливість використання математичних методів для дослідження формул алгоритмів.

Дослідженням [12] започаткований метод опису алгоритмів, який базується на ідеї створення операцій впорядкувань за індексами. Згодом розвинено його до прикладної алгебри секвенційних алгоритмів [13]. Багатолітня практика застосування прикладної алгебри секвенційних алгоритмів у різних областях показала доцільність спрощення алгоритмів, яким забезпечується зменшення затрат на їх реалізацію. Результати практики використання стали підставою для подальшого розвитку прикладної алгебри секвенційних алгоритмів, предметом якого і є дана стаття.

2. ОДНООСНОВНА АЛГЕБРА АЛГОРИТМІВ ПЕРШОГО ПОРЯДКУ

Систему позначень утворюють: знаки операцій: $\frown$ – логічного секвентування; $\vdash$ – логічного елімінування; $-$ – реверсування; $\sqcap$ – логічного паралелення; логічні змінні $x,y, \ldots, y_0, \ldots, u, u_j, u^i_j, \ldots$; значення змінних 0 та 1; * – порожній знак; розділювачі змінних та їхніх значень (кома, крапкою з комою і двокрапка); індекси порядку α та β.

Означення операцій секвентування, паралелення і реверсування над логічними значеннями і змінними подано у нижче наведеній таблиці 1. Таблицю 1 побудовано для індексів α та β, якими індексовано змінні x та y. Логічне секвентування змінних x та y, які мають індекс (індексовано α) записано як $\overset{\frown}{x_\alpha, y_\alpha}$. З метою спрощення запису логічного секвентування індекси опускатимо. У такому разі логічне секвентування $\overset{\frown}{x_\alpha, y_\alpha}$ запишеться у вигляді $\overset{\frown}{x,y}$. Якщо ж змінні матимуть не однакові індекси, наприклад, $\overset{\frown}{x_\alpha, y_\beta}$, то розділятимо їх крапкою з комою, що для наведеного прикладу дає вираз $\overset{\frown}{x;y}$. Аналогічні спрощення використовуємо і в операції логічного паралелення. Наприклад, вирази логічного паралелення з індексованими змінними $\overline{x_\alpha, y_\alpha}$, $\overline{x_\alpha, y_\beta}$, $\overline{x_\alpha, y_\beta}$ та $\overline{\overline{x_\alpha, y_\alpha}}$ запишемо як $\overline{x,y}$; $\overline{x;y}$; $\overline{\overline{x;y}}$ та $\overline{\overline{x,y}}$.

Як видно із таблиці 1 (Зм. – скорочення слова змінні та індекси, а № – порядкові номери комбінацій значень змінних) операція логічного секвентування набуває значення 1 тільки у тому випадку коли значеннями обох змінних є 1. Вона є комутативною тільки у випадку змінних з одинаковими індексами порядку. У загальному випадку операція логічного секвентування є некомутативна.

З таблиці 1 теж видно, що операція логічного паралелення у загальному випадку є некомутативною і набуває значення істинності 1 тільки тоді, коли хоча б одна із її змінних має значення істинності 1.

Реверсним до значення істинності 1 є значення 0 і навпаки – реверсним значенням 0 є 1.

Таблиця 1

Істинність операцій логічного секвентування, паралелення і реверсування.

Зм. №	Індекс α		Індекс β		$\overset{\frown}{x_\alpha,y_\alpha}$	$\overset{\frown}{x_\alpha,y_\beta}$	$\overset{\lceil\ \rceil}{x_\alpha,y_\alpha}$	$\overset{\lceil\ \rceil}{x_\alpha,y_\beta}$	$\overline{\overset{\lceil\ \rceil}{x_\alpha,y_\beta}}$	$\overline{\overset{\lceil\ \rceil}{x_\alpha,y_\alpha}}$	$\overset{\frown}{y_\alpha,x_\beta}$
	Змінні		Змінні								
	x_α	y_α	x_β	y_β							
0	0	0	0	0	0	0	0	0	1	1	0
1	0	0	0	1	0	0	0	1	0	1	0
2	0	0	1	0	0	0	0	0	1	1	0
3	0	0	1	1	0	0	0	1	0	1	0
4	0	1	0	0	0	0	1	0	1	0	0
5	0	1	0	1	0	0	1	1	0	0	0
6	0	1	1	0	0	0	1	0	1	0	1
7	0	1	1	1	0	0	1	1	0	0	1
8	1	0	0	0	0	0	1	1	0	0	0
9	1	0	0	1	0	1	1	1	0	0	0
10	1	0	1	0	0	0	1	1	0	0	0
11	1	0	1	1	0	1	1	1	0	0	0
12	1	1	0	0	1	0	1	1	0	0	0
13	1	1	0	1	1	1	1	1	0	0	0
14	1	1	1	0	1	0	1	1	0	0	1
15	1	1	1	1	1	1	1	1	0	0	1

Означення операції елімінування над логічними змінними наведено у таблиці 2.

Таблиця 2

Операція логічного елімінування над істинністними змінними

Зм. №	x	y	u	$\vdash x;\ y;\ u\text{-}?\dashv$	$\vdash x;\ y;\ 0\text{-}?\dashv$	$\vdash x;\ y;\ 1\text{-}?\dashv$	$\vdash x;\ y;\ *\text{-}?\dashv$
0	0	0	0	0	0	0	$\lceil 0,0 \rceil$
1	0	0	1	0	0	0	$\lceil 0,0 \rceil$
2	0	1	0	1	1	0	$\lceil 0,1 \rceil$
3	0	1	1	0	1	0	$\lceil 0,1 \rceil$
4	1	0	0	0	0	1	$\lceil 1,0 \rceil$
5	1	0	1	1	0	1	$\lceil 1,0 \rceil$
6	1	1	0	1	1	1	$\lceil 1,1 \rceil$
7	1	1	1	1	1	1	$\lceil 1,1 \rceil$

Аксіоматичне означення операцій.

Означення 1. Операція, яка має властивості:

- ідемпотентності: $\overset{\frown}{x,\ x}=x;$
- комутативності: $\overset{\frown}{x,y}=\overset{\frown}{y,x};$

- утворення логічного значення: $\overbrace{x, \bar{x}}=0$;
- виелімінування операції: $\overbrace{x; 0}=0, \overbrace{0; x}=0$;
- асоціативності: $\overbrace{\overbrace{x, y}, z}=\overbrace{x, \overbrace{y, z}}$;
- отримання змінної: $\overbrace{x, 1}=x, \overbrace{x; 1}=x,$

називатимо *логічним секвентуванням*.

Означення 2. Операція, яка має властивості:
- ідемпотентності: $\boxed{x, x}=x$;
- комутативності: $\boxed{x, y}=\boxed{y, x}$;
- утворення логічного значення: $\boxed{x, \bar{x}}=1$;
- виелімінування операції: $\boxed{x, 1}=1, \boxed{x; 1}=1, \boxed{1; x}=1$;
- виелімінування значення: $\boxed{x, 0}=x$;
- асоціативності: $\boxed{\boxed{x, y}, z}=\boxed{x, \boxed{y, z}}$;
- дистрибутивності: $\overbrace{x, \overbrace{y: z}}=\overbrace{\overbrace{x, y}: \overbrace{x, z}}$; $\overbrace{\overbrace{y: z}, x}=\overbrace{\overbrace{y, x}: \overbrace{z, x}}$;

називатимо *логічним паралеленням*.

Означення 3. Операція, яка має властивості:
- отримання з логічного секвентування логічного паралелення: $\overline{\overbrace{x: y}}=\overbrace{\bar{x}: \bar{y}}$;
- перетворення логічного значення $0: \bar{0}=1$;
- подвійного перетворення: $\bar{\bar{x}}=x,$

називатимо *логічним реверсуванням*.

Означення 4. Операція, яка має властивості:
- вибору змінної за умовою: $\vdash x; y; u\text{-}? \dashv = \overbrace{\overbrace{u; x}, \overbrace{\bar{u}; y}}$;
- ідемпотентности: $\vdash x; x; u\text{-}? \dashv = x$;
- вибору умови: $\vdash x;y;u_1\text{-}? \dashv ; \vdash x;y;u_2\text{-}? \dashv ; u_3\text{-}? = \vdash x;y;\vdash u_1\text{-}?;u_2\text{-}? \dashv ;u_3\text{-}? \dashv$;
- дистрибутивности: $\vdash \overbrace{x,y;x,z} ;u\text{-}? \dashv = \overbrace{x,y;z;u\text{-}?}, \overbrace{y,x;\ z,x;u\text{-}?} = \overbrace{y;z;u\text{-}?},x;$
- поглинання змінної: $\vdash x;y;u\text{-}? \dashv ; z;u\text{-}? = x;z;u\text{-}?, \vdash x;y;z;\ u\text{-}?;u\text{-}? \dashv = x;z;u\text{-}?,$

називатимо *логічним елімінуванням*.

Означення 5. Змінні або їхні значення з приписаними їм операціями секвентування, елімінування і паралелення індексами називатимо *унітермами*.

Означення 6. *Формулою* є тільки той вираз, для якого це можна показати, застосувавши скінчену кількість раз пункти 1– 3:

1. Якщо x та y – значення, змінні або унітерми, то $\overline{x:y}$ і $\overline{x:y}$ – формули.

2. Якщо x, y, а u – умова, то $\overline{x; y\ ; u\text{-}?}$, $\overline{y\ ; x\ ; u\text{-}?}$ – формули.

3. Якщо x, y – значення, змінні, унітерми або формули, а u – умова, то

$\overline{x}$, $\overline{y}$, $\overline{\overline{x: y}}$, $\overline{\overline{x: y}}$ і $\overline{x; y; u\text{-}?}$ та $\overline{y; x; u\text{-}?}$ – формули.

Одноосновною алгеброю алгоритмів першого порядку є система логічних операцій секвентування, паралелення, елімінування та реверсування над унітермами-значеннями і унітермами-змінними з можливістю переназивання та заміни унітермів-змінних з даним індексом, всюди де вони входять з цим індексом у формулу, у якій виконується заміна, унітермами-значеннями чи будь-якими логічними операціями або утвореними із них виразами.

Проілюструємо прикладами використання одноосновної алгебри алгоритмів для опису динамічних форматів структур даних відомої інформаційної системи опрацювання формул алгоритмів [14].

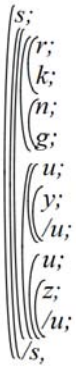
Приклад 1. Побудуємо модель формату даних унітерму, яка використовується для запису і зберігання у пам’яті комп’ютера та зчитування з неї значень унітермів. Для виконання зчитування унітермів з комп’ютерної пам’яті необхідним є наявність ідентифікатора за яким вони будуть розпізнавані. Оскільки іде мова про унітерми, то цілком логічно назвати цю змінну u. Комп’ютерна пам'ять, а також можливості монітора є обмеженими, тому доцільним є обмеження кількості знаків, якими будуть утворені самі унітерми. Доцільність уведення обмежень на кількість знаків унітермів також обумовлена можливостями людського сприйняття і наочністю. У відомій інформаційній системі опрацювання формул алгоритмів [14] кількість знаків унітермів не перевищує 256. Отже будь-який унітерм (x) може мати від нуля до 256 знаків унікоду. Наявність обмеження максимальної кількості знаків, але відсутність фіксованої кількості знаків кожного з унітермів зокрема, з метою знаходження їхнього кінця, потребує уведення ознаки кінця унітермів. У ролі такої ознаки доцільним є уведення змінної $/u$. Отож формат структури даних унітермів формул алгоритмів утворений трьома складовими u, x і $/u$. У форматі даних унітермів його ідентифікатор є першим, сам унітерм – другим, а ознака кінця унітерму – третьою. Їхня перестановка недопустима. Вони утворюють послідовність унітермів, яку опишемо операцією секвентування з розділювачом їх крапка з комою

$$\overline{u; \overline{x; /u}}.$$

Змінна u набуває значення 1, якщо значенням комірки пам’яті p_u, яка перевіряється на наявність ідентифікатора опису унітерма є un,

інакше ця змінна набуває значення 0. Значенням змінної *x* є 1, якщо у комірках пам'яті p_x між *u* і */u* кількість знаків унітерму *t* не перевищує 256. В іншому разі її значенням є 0. Для змінної */u* отримуємо значення 1, якщо значенням комірки пам'яті $p_{/}$ є */un*, інакше – 0.

Приклад 2. Модель формату даних операції секвентування має такий вигляд



де *s* – змінна-ідентифікатор першого поля (p_s) структури даних операції секвентування, яка набуває значення 1, якщо це поле містить значення *seq* і значення 0 – в усіх інших випадках; *r* – змінна-ідентифікатор знаку розділювача унітермів, яка є 1, якщо її поле p_r має значення *sep*, в усіх інших випадках її значенням є 0; *k* – логічна змінна зі значенням 1, коли поле розділювача унітермів p_k операції секвентування містить значення *"com"* або *"sem"*, в усіх інших випадках вона набуває значення 0; *n* – змінна-ідентифікатор орієнтації знаку операції секвентування, значенням якої є 1, коли її поле p_n містить значення *ori*, інакше – вона набуває значення 0; *g* – змінна, яка набуває значення 1 тільки у тому випадку, коли у полі значення орієнтації p_g зберігається слово *"hor"* або *"ver"*; *y* і *z* – змінні, відповідні полям p_y і p_z у яких зберігаються значення двох унітермів операції секвентування; */s* – ідентифікатор кінця опису формату даних операції секвентування є 1, коли відповідне їй поле $p_{/s}$ містить текст */seq*, а в усіх інших випадках її значенням є 0.

3. ОДНООСНОВНА АЛГЕБРА АЛГОРИТМІВ ДРУГОГО ПОРЯДКУ

Систему позначень утворюють: – знаки алгебри алгоритмів першого порядку; – логічні ***функціональні*** змінні *F(a), R(a, b)*, … , значеннями яких є 0 і 1 та які залежать від однієї або декількох ***індивідних*** змінних *a, b*, … ; – знаки операцій циклічного секвентування $\not<$, циклічного елімінування і циклічного паралелення $\not>$; – рівності =.

Відома операція ***дорівнює*** з аксіомами: $x=x$, якщо $x=y$ то $y=x$, якщо $x=y$, а $y=z$ то $x=z$.

Означення 7. Операцію з властивостями:

реверсування: $\overline{\not<xF(x)}=\not>x\overline{F(x)}$;

- секвентування: $\not<xF(x)=\overline{F(i):\overline{F(j):\overline{F(k):\ldots}}}$, для $x\in Q=\overline{i:\overline{j:\overline{k:\ldots}}}$ (індивідні змінні, які знаходяться під знаками операцій опису циклів називатимо *зв'язаними* цими операціями, а не зв'язані змінні називатимо *вільними*, змінна x є звязаною);

– порожнього циклу: $\not<x^*x = *$,

називатимо *циклічним логічним секвентуванням*.

Означення 8. Операцію з властивостями:

- реверсування: $\overline{\not>xF(x)}= \not< x\overline{F(x)}$;

- паралелення: $\not>xF(x)=\overline{F(i):\overline{F(j):\overline{F(k):\ldots}}}$, для $x\in Q$;

- порожнього циклу: $\not>x^*x=*$,

називатимо *циклічним логічним паралеленням*.

Означення 9. Операцію з властивостями:

- елімінування: $\not>u_xF(x)=\overline{F(i);\ \overline{F(j);\ \overline{F(k);\ldots u_k\text{-}?;u_l\text{-}?;u_l\text{-}?}}}$, для $x\in Q$;

- порожнього циклу: $\not>x^*_x = *$,

називатимо *циклічним логічним елімінуванням*.

Означення 10. Логічні функціональні змінні, які впорядковані логічними операціями секвентування, елімінування, паралелення або (і) логічними операціями циклів, називатимо логічними *функціональними унітермами*.

Означення 11. Формулами є вирази одноосновної алгебри алгоритмів першого порядку, а також вирази, для яких це можна показати, застосувавши скінчену кількість раз пункти 1- 5:

1. Якщо *D, K* – значення, змінні, функціональні унітерми або формули, то $=(D,K)$ і $D=K$ – формули.

2. Якщо *F(a,b, ...)* та *P(c,d, ...)* – логічні функціональні унітерми, які залежать від однієї або декількох індивідних змінних, то $\overbrace{F(a,b,...):P(c,d,...)}$ і $\overline{F(a,b,...):P(c,d,...)}$ – формули.

3. Якщо *A(a,b,...)*, *B(c,d,...)* – логічні функціональні унітерми або алгоритми, а *u* – унітерм умови, то $\overline{A(a,b,...);B(c,d,...);u\text{-}?}$, $\overline{B(c,d,...);A(a,b,...);u\text{-}?}$ – формули.

4. Якщо *A(a,b,...)*, *B(c,d,...)* – функціональні унітерми або алгоритми, а *u* – унітерм умови, то $\overline{A(a,b,...)}$, $\overline{B(c,d,...)}$, $\overline{\overline{A(a,b,...):B(c,d,...)}}$, $\overline{\overline{A(a,b,...):B(c,d,...)}}$ і $\overline{\overline{A(a,b,...);B(c,d,...,);u\text{-}?}}$, $\overline{\overline{B(c,d,...);A(a,b,...);u\text{-}?}}$ – формули.

5. Якщо *x, a, ...* – змінні, *F(x:a: ...)* є функціональним унітермом або алгоритмом, то $\overline{\subset xF(x,a,...)}$, $\overline{\not\subset xF(x,a,...)}$, $\overline{\emptyset xF(x,a,...)}$, $\overline{\not\emptyset xF(x,a,...)}$, $\overline{\subset xF(x,a,...)}$, $\overline{\not\subset xF(x,a,...)}$ – формули.

Одноосновною алгеброю алгоритмів другого порядку є система логічних операцій секвентування, паралелення, елімінування, реверсування, циклічних секвентування, паралелення, елімінування над унітермами-значеннями і унітермами-змінними та логічними функціональними унітермами з можливістю переназивання та заміни унітермів-змінних, індивідних змінних іншими індивідними змінними і логічних функціональних унітермів з даним індексом, всюди де вони входять з цим індексом у формулу, у якій виконується заміна, унітермами-значеннями чи будь-якими логічними операціями або утвореними із них виразами.

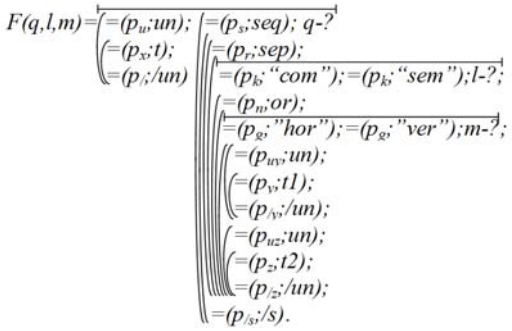
Приклад 3. Проілюструємо використання одноосновної алгебри алгоритмів другого порядку алгоритмом формування структур даних унітерма і операції секвентування. Для того щоб ідентифікувати формат структури даних, яким може бути формат унітерму чи операції секвентування, уводимо вхідну змінну *q*. Заложімо, що її значення 1 означає формування структури даних унітерму, а значення істинності 0 – формату даних операції секвентування.

Розділювачом двох унітермів операції секвентування може бути кома чи крапка з комою. Заложімо, що тип розділювача задається вхідною змінною *l*, а її значення 1 означає, що розділювачом є кома і 0 – крапка з комою.

Знак операції секвентування має горизонтальну чи вертикальну орієнтацію. Заложімо же тип орієнтації задається вхідною змінною *m*, а її значення 1 означає задання горизонтальної орієнтації, тоді як 0 – задання вертикальної орієнтації.

У такому разі алгоритм формування структур даних (*F*) має три вхідних змінних і в загальному вигляді його запишемо як *F(q,l,m)*. Він

починається розпізнаванням значення істинності вхідної змінної q, що виражено елімінуванням за цією змінною. Функціональний унітерм $=(p_u;un)$ набуває значення істинності 1, якщо значення змінної p_u є un, інакше – він має значення істинності 0. Аналогічно визначаються значення істинності усіх решти функціональних унітермів, а сама формула алгоритму має такий вигляд



В елімінуванні за логічним значенням вхідної змінної l визначається тип розділювача унітермів операції секвентування. Орієнтація знаку операції секвентування розпізнається в елімінуванні за логічним значенням змінної m. $t1$ і $t2$ є значеннями унітермів операції секвентування.

4. ПІДСУМОК

Означеними одноосновними алгебрами алгоритмів першого і другого порядків описуються структури даних та алгоритми інформаційних технологій і систем.

1. Kleene S.C.: Origins of recursive function theory. Annals of the Theory of Computing, vol. 3, No. 1, Jan. 1981, – P. 52-67. 2.Church A.: An unsolvable problem of elementary number theory. American Journal of Mathematics, vol. 58 (1936), – P. 345-363. 3.Turing A. M.: On computable numbers, with an application to the Entscheidungsproblem. Proceedings of London Mathematical Society, series 2, vol. 42 (1936-1937), – P. 230-265. 4.Post E. L. Finite Combinatory Processes – Formulation 1. Journal of Symbolic Logic, 1, 1936. -P. 103 -105. 5.Марков А.А. Теория алгорифмов //Труды МИАН. – Т.38. 1951. – С. 176-189. 6.Колмогоров А.Н. О понятии алгоритма //УМН. – Т.8, вып. 4 (56). 1953. – С. 175-176. 7.Schönhage A.: Universelle Turing Speicherung. In J. Dörr and G. Hotz, Editors, Automatentheorie und Formale Sprachen, Bibliogr. Institut, Mannheim, 1970. – P. 369-383. 8.Aho A.V, Hopcroft J.E, Ullman J.D.: The design and analysis of computer algorithms. Addison-Wesley Publishing Company, 1974. 9.Криницкий Н.А. Алгоритмы вокруг нас. – М.: Наука. 1984. – 224 с. 10.Цейтлин Г.Е. Введение в алгоритмику. – К.: Сфера, 1998. – 310 с. 11.Цейтлин Г.Е., Захария Л.М. Алгеб-

раические аспекты полноты: абстракции, биология и экология //Проблеми програмування. 2008. № 2-3. – С.31 – 36. 12.Овсяк В.К. Программа имитации функционирования многозначных логических структур (УПОРЯДОЧЕНИЯ). № АП0159. – К.: Укр. РФАП, 1987. – 450 с. 13.Овсяк В.К. Засоби еквівалентних перетворень алгоритмів інформаційно-технологічних систем //Доповіді Національної академії наук України, № 9, 1996. – С.83-89. 14.Овсяк О.В. Модель інформаційної технології формування операції секвентування /О.Овсяк //Вісник Національного університету "Львівська політехніка": Комп'ютерні науки та інформаційні технології. – № 694, 2011. – С. 166 – 173.