УДК 004.043

Гамаюн В.П., д-р техн. наук
Андрєєв А.А.

ОБЧИСЛЕННЯ СТЕПЕНЕВИХ ФУНКЦІЙ ТА ФАКТОРІАЛУ З ЗАСТОСУВАННЯМ РОЗРЯДНО-ЛОГАРИФМІЧНОЇ АРИФМЕТИКИ

**Інститут комп'ютерних технологій
Національного авіаційного університету**

Розглянуто моделі обчислення функцій факторіалу та степеню з натуральним показником при розрядно-логарифмічному кодуванні даних, яке забезпечує обробку у великому діапазоні чисел та значне зменшення впливу округлення. Запропоновано підвищити швидкість обрахунку факторіалу за рахунок виділення похідних пар та чисел, що є степенями двійки. Описано використання схеми Горнера при розрядно-логарифмічному кодувані, та метод черг для прискорення операції множення, яка є складовою обчислення факторіалу та степеню

Вступ

Для сучасного етапу розвитку науки характерне розв'язання задач, що вимагають обчислень підвищеної точності, а отже і значних ресурсно-часових затрат. Такі задачі часто передбачають дослідження залежностей, які наближено або точно виражаються через елементарні функції. Інваріантними характеристиками при виборі розрахункових алгоритмів є швидкість та точність. Практичні задачі систем автоматичного проектування, криптографії, інформаційної безпеки, статистичного аналізу експериментальних даних та інших галузей використання комп'ютерних технологій часто вимагають безпомилкових або майже безпомилкових обчислень. Такі умови вимагають більш досконалих стандартів представлення числових даних, підвищеної точності розрахунків, мінімізація впливу обмежень розмірів розрядної сітки та методів обчислення.

Остаточно систему алгоритмів та методів визначають за системним підходом, який визначається наступними основними параметрами: час обчислення, апаратурні та програмні затрати, точність обчислень, діапазон зміни аргумента та можливість продовжувати обчислення при зростанні розрядності.

Постановка проблеми

На даний час найбільш широко використовується стандарт представлення чисел запропонований Інститутом Інженерів з Електроніки та Електротехніки (*Institute of Electrical and Electronics Engineers – IEEE*). Стандарт *IEEE*-754 [5], оновлений 2008 року, описує формати чисел, основні алготитми обчислень, правила округлень та обміну даних з плаваючою точкою. Та навіть на даний момент цей стандарт повноцінно не введенов експлуатацію і використовувати найточніший 128-розрядний формат можливо через апаратну або програмну емуляцію. Таким чином актуальними на сьогоднішній момент залишаються формати з розрядністю 32 та частково 64 біти.

Багаторозрядна арифметика, що пропонується як альтернативна до відомої двійкової арифметики має ряд переваг при реалізації для комп'ютерної техніки серійного виробництвах [1]. І в ході реалізації елементарних функцій, зокрема факторіалу та степеню числа, виникли наступні задачі:

– порівняння точності відображення результатів стандартної та альтернативної арифметик;

– збереження доступної точності обчислень;

– зменшення часу обчислень.

Методика досліджень

Зі стандарту *IEEE*-754 розглянемо найбільш точний формат ***binary*128** (рис. 1).

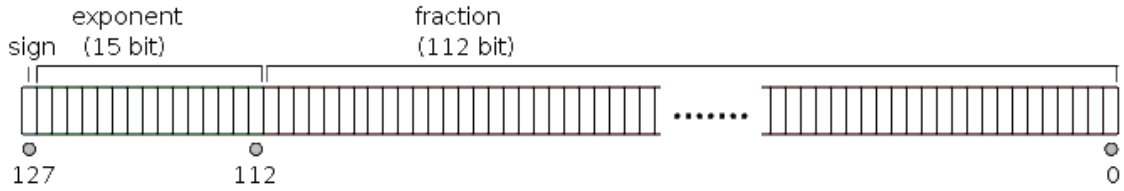

Рис. 1. Формат *binary*128 числа з плаваючою точкою

Даний формат, як і його попередники (*binary*16, *binary*32 та *binary*64), передбачає неявно заданий додатковий біт мантиси. Цей біт є старшим бітом і завжди дорівнює одиниці. Таким чином в пам'яті зберігається 112 біт, а насправді число зображає 113, а це відповідає приблизно 34 десяткових знаки:

$$N = \log_{10}(2^{113}) \approx 34{,}016 . \qquad (1)$$

Тобто, якщо результат обчислень, введене значення, або будь-яким іншим чином отримане число займатиме в мантисі 113+*k* розрядів, то молодші k розрядів буде втрачено. Формати *binary*32 та *binary*64 передбачають довжину мантиси 24 (23+1) та 53 (52+1) двійкових розряди, що відповідає 7-8 ($N = \log_{10}(2^{24}) \approx 7{,}225$) та 15-16 ($N = \log_{10}(2^{53}) \approx 15{,}955$) десятковим знакам.

Розрядно-логарифмічна система числення забезпечує розширення діапазону данних та обробку з використанням арифметики логарифмів при тій самій розрядності. Така система є розширенням двійкової системи числення за умов кодування кожного ненульового розряду двійкового операнду. Кожен код дорівнює значенню логарифму від ваги цього ненульового розряду:

$$N_i = \log_2 a_i p^i = i(a_i \neq 0), \qquad (2)$$

де a_i ={0,1} – цифра двійкового числа, p_i = 2 – основа системи числення, в даному випадку двійкової. Іншими словами число *A* (деякий числовий операнд) може бути представлений в вигляді масиву (вектора) двійкових кодів:

$$N_1 N_2 N_3 ... N_i ... . \qquad (3)$$

Наприклад, двійкове число $A_2 = 11010001.0101$ в розрядно-логарифмічному представлені матиме наступний вигляд:

$$A_{ÐÊ} = 7.6.4.0.-2.-4 .$$

В ході досліджень було виявлено, що при програмній реалізації розряди при такому кодуванні раціональніше розміщати в пам'яті комп'ютера за зростанням, тобто $A_{РЛ} = -4.-2.0.4.6.7$.

Розрядна сітка розміром 32 згідно стандарту *IEEE*-754 може представити цілі числа з діапазону від -2^{31} до $+2^{31}$, та числа з плаваючою точкою з діапазону від 1,2 10^{-38} до 3,4 10^{+38} та діапазону симетричного відносно нуля. Розрядно-логарифмічне (РЛ) кодування для розрядної сітки того ж розміру дозволяє представити числа з діапазону від $-2^{2^{31}-1}$ до $+2^{2^{31}-1}$ з точністю $2^{-2^{31}+1}$, до того ж РЛ-кодування представляє єдиний формат для цілих та дробових чисел.

В [1] операція ***множення*** чисел *A* та *B* описується як процедура, що складається з наступних етапів:

– кожен елемент вектора *A* сумується з кожним елементом вектора *B*, кожна така сума є розрядом добутку, фактично формується масив (4) часткових добутків;

– впорядкування елементів добутку;

– серед елементів добутку виконується зведення подібних – кожні два од-

накові розряди замінюються одним, збільшеним на одиницю.

	A0	A1	…	An
B0	A0+B0	A1+B0	…	An+B0
B1	A0+B1	A1+B1	…	An+B1
…	…	…	…	…
Bm	A0+Bm	A1+Bm	…	An+Bm

(4)

Саму ж операцію множення в розрядно-логарифмічних кодах можна прискорити, використовуючи властивість градієнтності [2] масиву часткових добутків. Мінімальний та максимальний елементи проміжного результату знаходяться на кінцях головної діагоналі, тобто елементи матриці поступово зростають в напрямку від верхнього лівого кута до нижнього правого. Таке розміщення і називається *градієнтністю*. Прискорення досягається зміною порядку обрахунку елементів матриці (4) з рядків на діагоналі, паралельні побічній, що дає можливість уже на етапі обрахунку елементів використовувати поелементне «зведення елементів» та сортування.

За іншим методом прискореного множення масив (4) розглядається як сукупність *m* черг – кожен рядок є чергою, верхні рядки є молодшими чергами, а нижні – старшими:

– для визначення наступного розряду результату порівнюються чергові елементи з усіх черг і обираються два найменші – *k* та *p* ($k \le p$). Елемент з черги *k* додається до результату, а сама черга зсувається на 1 елемент;

– новий елемент з черги *k* порівнюється з *p*, молодший з них додається до вектора результату, а його черга зсувається на 1 елемент. Якщо молодшим виявився *k*, то повторюється даний пункт, якщо *p*, то попередній. Якщо $k=p$, то зсуваються обидві черги, до результату додається розряд, що дорівнює $k+1$ і виконується попередній пункт.

Коли одна з черг вичерпується, вона повністю виключається. Враховуючи впорядкованість масивів обох операндів, можна з впевненістю сказати що вичерпуються черги від молодшої до старшої, а тому віслідковувати всі немає необхідності.

Метод черг дозволяє повністю виключити сортування, а з мінімальною модифікацією може використовуватись як багатооперандне додавання.

Обчислення факторіалу

При розрахунках в задачах з теоретичної і статистичної фізики є нормою використання таблиць логарифмів факторіалів, та формули Стірлінга:

$$n! \approx \sqrt{2\pi n}\left(\frac{n}{e}\right)^n. \qquad (5)$$

Але для важливих задач експериментальної фізики, а також ряду задач криптографії, кодування та багатьох наукових досліджень необхідно мати змогу знаходити факторіал абсолютно точно і не за модулем, а цілком. Таке значення можна отримати лише прямим множенням. Як видно з рис. 2, формат *binary*128 здатен коректно відобразити факторіали чисел не більше 37, для *binary*64 – 22, для *binary*32 – 13. Тому для задач де є необхідність отримання точного значення факторіалу рекомендується використовувати розрядно-логарифмічну арифметику.

Розв'язуючи проблему прискорення обчислень пропонується для зменшення кількості множень виключити з черги множників всі степені двійки і замінити на відповідний зсув. Такий підхід є відомим, але при РЛ арифметиці, при розрядно-логарифмічному кодуванні, *зсув* числа виконується додаванням значення зсуву до кожно з розрядів числа. Тобто не має потреби у значних апаратних витратах. Для числа *n*, факторіал якого шукається, виключаються степені числа 2 і тому кількість множень зменшиться на

$$K = \lceil \log_2 n \rceil. \qquad (6)$$

Для РЛ системи кодування не потрібно обчислювати логарифм та виділяти цілу частину, тому що K в даному випадку дорівнює старшому розряду числа n. Зсув S дорівнюватиме сумі чисел від 1 до K:

$$S = \sum_{i=1}^{K} i \,. \qquad (7)$$

Розглямено чергу множників без викреслених чисел:

$$3 \cdot 5 \cdot 6 \cdot 7 \cdot 9 \cdot 10 \cdot 11 \cdot 12 \cdot 13 \cdot 14 \cdot 15 \cdot 17 \cdot \ldots \cdot \mathrm{n} \,. \qquad (8)$$

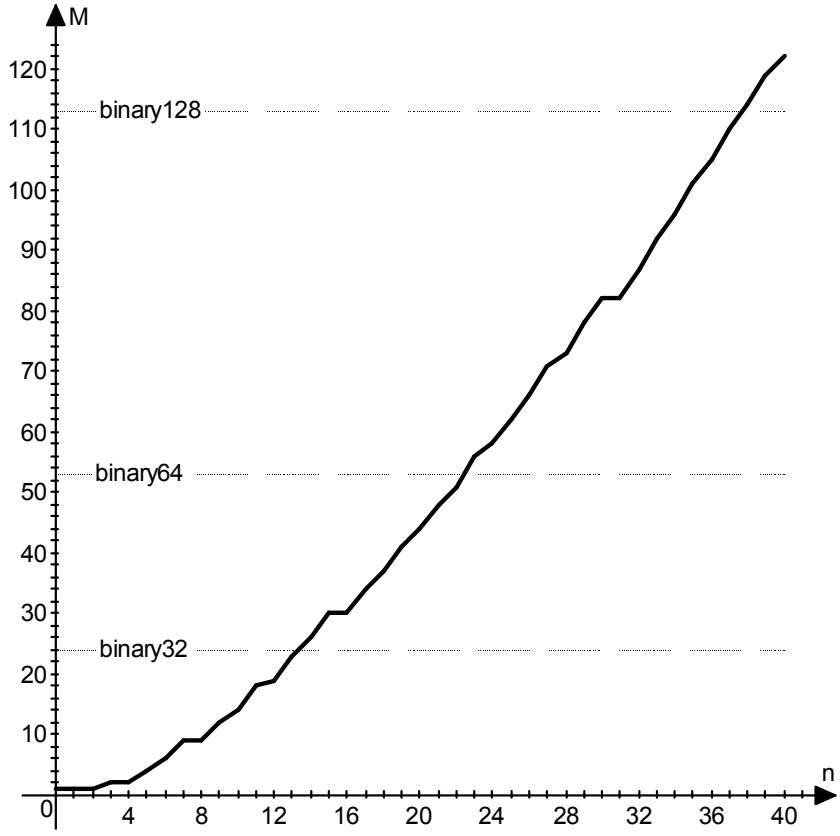

Рис. 2. Графік залежності необхідної довжини мантиси від числа, факторіал якого шукається

Використовуючи властивість транзитивності операції множення, перепишемо цей ряд наступним чином:

$$(3 \cdot 5) \cdot (6 \cdot 10) \cdot (12 \cdot 20) \cdot \ldots \cdot (7 \cdot 9) \cdot (14 \cdot 18) \cdot \ldots \cdot \mathrm{n} \,. \qquad (8)'$$

Оскільки

$$\left(3 \cdot 2^m\right) \cdot \left(5 \cdot 2^m\right) = (3 \cdot 5) \cdot 2^m \cdot 2^m = (3 \cdot 5) \cdot 2^{2m}, \qquad (9)$$

то після обчислення (3×5) немає необхідності обчислювати (6×10), (12×20) і т.д. Пару (3×5) називатимемо *першою парою*, а всі пари типу $\left(3 \times 2^m\right) \times \left(5 \times 2^m\right)$ *похідними парами* від першої. Аналогічно для інших пар. Таким чином можливо значно зменшити кількість множень. Далі для ряду (8)′ можна визначати кожну похідну пару за формулою (9), заміняючи множення на 2^{2m} зсувом на *2m*, або ж знайти необхудну степінь першої пари, а суму всіх зсувів додати до *S*. Для прикладу розглянемо перші три пари ряду (8)′:

$$\begin{aligned} &(3 \cdot 5) \cdot (6 \cdot 10) \cdot (12 \cdot 20) = \\ &= (3 \cdot 5) \cdot (3 \cdot 5) \cdot 2^2 \cdot (3 \cdot 5) \cdot 2^4 = \\ &= (3 \cdot 5)^3 \cdot 2^4. \end{aligned}$$

Необхідно обчислити $(3 \times 5)^3$, а число 4 додати до загального зсуву. Даний варіант є раціональнішим, через меншу кількість операцій, а також через можливість пришвидшення обчислення степеню, про що мова піде нижче. В загально-

му ж виділення перших пар і пошук відповідних їм похідних має сенс поки не викреслено всі числа від 1 до $\lceil n/2 \rceil$.

Загальний метод обчислення факторіалу для розрядно-логарифмічного представлення даних реалізується виконанням наступних етапів:

a) викреслення з черги всіх чисел які є степенями двійки, та обчислення зсуву за формулою (7). Ознака того що число є стенем двійки в РЛ представленні є те що воно містить лише один розряд. На практиці такі числа навіть не заносяться в чергу і вона має вигляд (8);

b) обчислення першої пари, викреслення з черги всіх її похідних пар, та обчислення зсуву для цієї пари;

c) значення обчисленої пари заноситься в додаткову чергу, а значення зсуву додається до зсуву обчисленого в пункті a;

d) якщо перше з невикреслених чисел черги більше $\lceil n/2 \rceil$, то виконуємо наступний пункт, інакше – пункт b;

e) обчислюємо добуток елементів основної черги, та добуток елементів додаткової черги. Перемножуємо ці значення та отримуємо попередній результат;

f) остаточний результат отримується зсувом, який обчислювався в пунктах a - c. Зсув в розрядно-логарифмічному кодуванні має вигляд додавання значення зсуву до кожного елементу вектора.

З методу виключено етапи перевірки на рівність аргументу нулю, одиниці, перевірку на цілість та додатність, а також деякі інші логічні тести, які є у відомих методах.

Таким чином кількість множень можна зменшити в середньому на $\lceil n/4 \rceil$, що підтверджено експериментально.

Обчислення степеню

Серед методів [6] піднесення до натурального степеню, найраціональнішим вважається метод квадратування (*exponentiation by squaring*). Фактично алгоритм складений за методом квадратування зводиться до мультиплікативного аналогу схеми Горнера [7].

Розглянемо приклад – нехай деяке число a, як уже говорилось для РЛ-кодування не має різниці ціле число чи дробове, необхідно піднести до степеню b = 100. В двійковій системі число 100 матиме вигляд 1100100, а в РЛ відповідно 2.5.6. Результат R, оримується наступним чином:

$$R = \prod_{i=1}^{E(b)} r_i \ , \qquad (10)$$

де $E(b)$ – кількість розрядів при РЛ кодуванні, що відповідає кількості одиниць в двійковому записі числа b; r_i – обчислюється так: $\underbrace{(...((a)^2)^2...)^2}_{b_i}$, де b_i – i-тий розряд показника степеню в розрядно-логарифмічному представленні. Таким чином $r_1 = ((a)^2)^2$, $r_2 = (((r_1)^2)^2)^2$, $r_3 = (r_2)^2$, тобто загальна кількість множень необхідна для обчислення всіх r_i дорівнює найстаршому розряду показника b в розрядно-логарифмічному представленні. Загальна ж кількість множень, необхідна для обчислення a^b дорівнює

$$E(b)+b_E-1=3+6-1=8,$$

що значно менше ніж 100.

Алгоритм піднесення до степеню для розрядно-логарифмічного представлення даних реалізується виконанням наступних етапів:

a) якщо вектор степеню містить розряд 0, то $r_1=a$;

b) знаходимо r_i квадратуванням r_{i-1}. При цьому r_0 вважається рівним a;

c) обчислюємо добуток $R = r_1 \times r_2 \times r_3 \times ... \times r_E$.

З алгоритму виключено перевірку аргументу на рівність, нулю та одиниці, операцію визначення знаку, при від'ємному аргументі, та інші елементарні тести.

Альтернативою до описаних методів можуть стати паралельні обчислення з використанням алгоритму матричного множення [6]. Відповідно до цього алго-

ритму при обчисленні $R=a^b$ формується матриця з b однакових рядків, що є розрядами числа a. Таким чином кількість стовпців матриці дорівнює $E(a)$ – кількості розрядів a при РЛ кодуванні. Матриця розділяється на підматриці розмірності $b\times i$,де i=1,2,...b. Кількість матриць кожного типу визначається за формулою

$$H = \frac{E(b)!}{i!(E(b)-i)!}. \qquad (11)$$

Для кожної з підматриць наперед визначаються формули складових кінцевого результату. Формули залежать від значень b та $E(a)$, і визначаються за правилом, що відображає закони обчислення добутку чи степеню: розряди одного операнду не перемножуються між собою. Обчислення за цими формулами проводяться паралельно.

Так, наприклад, якщо операнд a складається з розрядів a_1, a_2, a_3, a_4, a_5 ($E(a)=5$), а степінь $b=4$, то основна матриця розміром 4×5 розділиться на підматриці 4×1 (тип I), 4×2 (тип II), 4×3 (тип III), 4×4 (тип IV):

$$\underset{(I)}{\begin{vmatrix} a_1 \\ a_1 \\ a_1 \\ a_1 \end{vmatrix}} \cdots \underset{(II)}{\begin{vmatrix} a_1a_2 \\ a_1a_2 \\ a_1a_2 \\ a_1a_2 \end{vmatrix}} \cdots \underset{(III)}{\begin{vmatrix} a_1a_2a_3 \\ a_1a_2a_3 \\ a_1a_2a_3 \\ a_1a_2a_3 \end{vmatrix}} \cdots \underset{(IV)}{\begin{vmatrix} a_1a_2a_3a_4 \\ a_1a_2a_3a_4 \\ a_1a_2a_3a_4 \\ a_1a_2a_3a_4 \end{vmatrix}} \cdots$$

Для підматриць типу I обчислення формул зводиться до суми всіх розрядів, що входять до стовпця, $4a_i$. Формули для підматриць типу II визначаються як всі варіанти з комбінацій елементів по стовпцях, за виключенням варіантів визначених для матриць типу I:

$$4(a_i+3a_j),\ 4(a_j+3a_i),\ 6(\ 2a_i+2a_j).$$

На рис. 3 зображено процес формування комбінацій елементів за формулою $4(a_i+3a_j)$ для підматриць типу II.

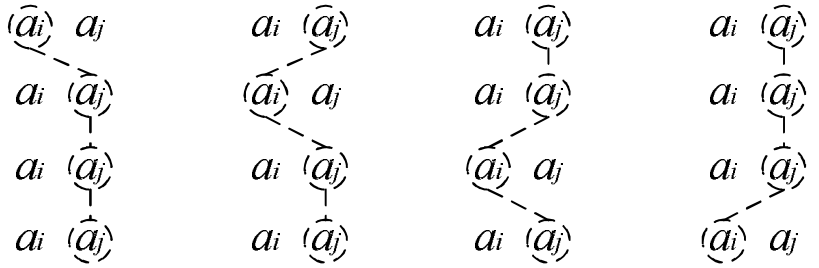

Рис. 3. Схема обчислення за формулою a_i+3a_j, $i\neq j$

Коефіцієнт 4 перед дужками означає кількість таких комбінацій. За аналогією визначаються формули для підматриць типів III та IV. Узагальнено формули для підматриці типу N (n стовпців) описати наступним правилом: в дужках знаходиться сума усіх n розрядів, що формують рядок підматриці; сума ненульових коефіцієнтів при цих розрядах дорівнює m – кількості рядків; множник перед дужками дорівнює кількості комбінацій елементів відповідно до частини формули в дужках.

Значення для всіх підматриць обчислюються паралельно і з них формується багаторядний код (набір складових). В позиціях такого коду записуються коефіцієнти, що розміщені в формулах перед дужками. Кінцевий результат визначається як сума.

Проведемо попередній порівняльний аналіз часових затрат для обрахунку проміжних розрядів результату за схемою Горнера та при паралельних обчисленнях.

Схема Горнера передбачає $E(b)$ ітеративних квадратувань числа a, що означає обрахунок $E(a)^{2E(b)}$ проміжних розрядів операцій множення. Кожен такий розряд при реалізації операції множення обчислюється як сума, тому загальний час розрахунків буде визначатись як

$$t_{Horner} \sim E(a)^{2E(b)} \cdot t_S, \qquad (12)$$

де t_S – час виконання операції додавання апаратним забезпеченням.

За описаний вище методом паралельного обчислення проміжні розряди результату розраховуються на основі підматриць в окремих блоках одночасно, тому загальний час розрахунків дорівнюватиме максимальному часу затраченому блоками. Формула для підматриць:

$$k(p_i a_i + p_j a_j + ... + p_z a_z). \qquad (13)$$

Якщо розмірність підматриці дорівнює $n \times b$, то в дужках знаходитиметься по n розрядів числа a, що відповідає n операціям множення та n-1 операціям додавання. При $f=n-b/2>0$ деякі з коефіцієнтів p (не менш ніж f) будуть рівними одиниці, тому максимальна кількість множень в формулі дорівнює $b/2$. Кількість додавань може асимптотично наближається до значення b. Прийнявши що операція множення займе вдвічі більше часу ніж додавання маємо:

$$t_{Mx} \sim \frac{b}{2} t_P + b t_S = 2 b t_S, \qquad (14)$$

де t_P – час операції множення. Порівнюючи формули (12) та (14), приходимо до висновку що паралельні обчислення для прямого розрахунку степеню при розрядно-логарифмічному кодуванні потенційно набагато швидші обчислень на основі ітераційних алгоритмів. Швидкість паралельних обчислень за описаним алгоритмом значною мірою залежить від комутаційних характеристик обислювальної техніки, та при достатній апаратній підтримці часові затрати на обчислення є мінімальними.

Висновки

Доведено ефективність використання розрядно-логарифмічного кодування в порівнянні з загальноприйнятими стандартами, для наукових досліджень, важливих розрахунків, задач криптографії, та ін. В результаті дослідженнь запропоновано альтернативні методи обчислення функцій факторіалу та піднесення до степеня, які є складовими розрахунку елементарних функцій. При розрахунках таких функцій зменшується кількість множень, а також описано метод черг, який дозволяє прискорити саме множення. Описано метод паралельного обчислення степеню, що базується на алгоритмі матричного множення.

Список літератури

1. *Гамаюн В.П.* Разрядно-логарифмическая арифметика, Методы и алгоритмы. – К.: Книжное издательство НАУ, 2007. – 272 с.

2. *Гамаюн В.П., Андреєв А.А., Чайка М.П.* Прискорення обчислення адитивно-мультиплікативних операторів при розрядно-логарифмічному кодуванні. – К: Видавництво НАУ, 2009. – 4 с.

3. *Захаров А.В., Хачумов В.М.* Разрядно-параллельные вычисления в системах реального времени. – Издательство «Университет города Переславля», 2003. – С. 97–104.

4. *Благовещенский Ю.В., Теслер Г.С.* Вычисление элементарных функций на ЭВМ. – Издательство «Техника», 1977. – 208 с.

5. http://en.wikipedia.org/wiki/IEEE_754-2008.

6. http://en.wikipedia.org/wiki/Exponentiation_by_squaring.

7. http://en.wikipedia.org/wiki/Horner_scheme.

Подано до редакції 29.04.10