**УДК 621.3**

**Микола Черкаський, Хусейн Халіл Мурад**
Національний університет "Львівська політехніка",
кафедра комп'ютерної інженерії

# КОМП'ЮТЕРНІ АЛГОРИТМІЧНІ СИСТЕМИ



**Розглянуто еволюцію тлумачення поняття "алгоритм", наведено різні підходи до його тлумачення. Останнє торкається комп'ютерного розв'язання задач. Формально визначена апаратно-програмна модель алгоритму (SH-модель), що уточняє поняття алгоритму стосовно комп'ютерних засобів. Розширено перелік властивостей алгоритму. До списку характеристик додано апаратну і структурну складності алгоритму. Програмна і структурна складності визначаються як кількість інформації, що міститься в комп'ютерних засобах.**

**Evolution of understanding the term "algorithm" is examined; some ways of understanding are presented. This one touches upon task solution by using a computer. Software/Hardware model of algorithm (SH-model) is formally defined, dears the definition of algorithm according to a computing means. List of properties of algorithm is expanded. Hardware and software complexity of algorithm are added into the list of properties. Hardware and software complexity are presented as information quantity, which is contained of computing.**

## Вступ

Алгоритм – фундаментальне поняття математики, його не можна математично точно визначити за допомогою інших фундаментальних понять. Тому користуються інтуїтивним тлумаченням. За останнє століття вербальне інтуїтивне тлумачення алгоритму набуло різноманітних форм [7, 10]. Наведемо два приклади тлумачень, які, на думку авторів, є найбільш характерними. Перше: "*Алгоритм – це послідовність інструкцій для виконання деякого завдання"* [10], друге: " *Алгоритм – це точний припис, який задає обчислювальний процес, що починається з деякого початкового даного і спрямований на одержання результату, який повністю відповідає цим початковим даним"* [6]. Ці формулювання не є математичними об'єктами. Вони не можуть бути основою для дослідження математичних проблем. Вони не можуть бути основою для розв'язання математичних задач, в тому числі для синтезу, аналізу та оптимізації комп'ютерних (технічних) об'єктів.

Дослідження алгоритмів проводять на моделях, надаючи їм точного математичного змісту. В літературі розглядаються моделі двох типів. До перших відносяться моделі, що дають змогу досліджувати побудову лише припису, послідовності інструкцій, що задають обчислювальний процес, наприклад, математичні функції, блок-схеми програм, програми на мовах високого рівня тощо. До моделей другого типу відносяться моделі, які, крім програми, включають також засоби перетворення початкових даних у результат. За такими моделями закріпилася назва – формальні алгоритмічні системи (ФАС). Прикладами абстрактних моделей алгоритмів ФАС є машина Тюрінга, нормальні алгоритми Маркова тощо. В основу побудови абстрактних моделей ФАС покладено *абстрактний обчислювач*, який "не враховує технічних подробиць" [6]. Ця особливість не дає змоги в повному обсязі впроваджувати здобутки теорії алгоритмів в практику побудови і дослідження апаратно-програмних засобів комп'ютерної техніки.

Сьогодні поняття алгоритму вийшло за рамки чистої математики. Воно набуло фундаментального змісту також у комп'ютерній техніці. На відміну від тлумачення алгоритму з

абстрактним обчислювачем, тлумачення алгоритму в комп'ютерних науках використовує апаратно-програмний обчислювач. У роботі описується модель алгоритму з використанням такого обчислювача (SH-модель) [8], розглядаються особливості SH-моделі порівняно з моделями на основі абстрактного обчислювача.

**Моделі алгоритмів**

Одну з перших моделей формальних алгоритмічних систем запропонував Алан Тюрінг. Вона отримала назву "машина Тюрінга". Машина Тюрінга уточняє інтуїтивне поняття алгоритму, вона має таке математичне визначення [5]:

Машина Тюрінга – це шістка

$$\langle A, Q, a_{0'}, q_0, q_1, P \rangle, \tag{1}$$

де $A$ – кінцева множина символів зовнішнього алфавіту; $Q$ – кінцева множина символів внутрішнього алфавіту; $q_0$ *і* $q_f$ – початковий і кінцевий стани, $q_0, q_f \in Q$; $a_o$ – позначення порожньої комірки стрічки, $a_0 \in A$; множини $A, Q$ не перетинаються і не містять літер $L, R, S$; $P$ – така програма, яка не може мати двох різних четвірок $P$, у яких би збігались два перші символи:

$$P = \{A\} \times \{Q\} \to \{A\} \times \{L, R, S\} \times \{Q\}, \tag{2}$$

де $L-$ зсувати головку вліво; $R-$ зсувати головку вправо; $S-$ залишити головку на місці

Машина Тюрінга повністю відповідає наведеному тлумаченню алгоритму, вона має всі його властивості і має вісім параметрів: правило початку, правило вводу, правило виводу, правило безпосереднього перероблення, правило закінчення, систему вхідних даних, потенційно нескінченну систему проміжних результатів, систему кінцевих результатів. Крок машини Тюрінга моделюється декількома "простими" операціями: читання символу зі стрічки, знаходження і читання команди в програмі, реалізація команд, запис або стирання нового символу, пересування головки вправо, вліво або залишання на місці. Перевагою над рекурсивними функціями з точки зору виконання вимоги елементарності є, крім зрозумілості підоперацій, також однотипність кроків для будь-яких алгоритмів. Фіксованість кроку машини Тюрінга дала змогу порівнювати різні алгоритми, а також сформулювати початкові визначення теорії складності – дати поняття часової, місткісної і програмної складностей [2].

Машина Тюрінга, як і більшість інших моделей ФАС (багато з них є варіантами машини Тюрінга, наприклад, машина з необмеженими регістрами), орієнтовані на дослідження обчислювальних операцій з використанням абстрактних обчислювачів. Абстрактні обчислювачі не враховують технічних подробиць виконання обчислювальних операцій. Така технічна характеристика, як апаратна складність, не існує для ФАС з абстрактним обчислювачем. Декларування технічних (апаратних) засобів, необхідних для перетворення даних, відсутнє у визначеннях моделей абстрактних ФАС. Це ми бачимо на прикладі машини Тюрінга. У формальному математичному визначенні машини Тюрінга відсутні дані про елементи її конструкції. Головка читання, стрічка, програма – все це елементи машини, яку не можна побудувати, але можна уявити. Уявний образ цієї машини використовується лише для пояснення принципу її роботи. Крок машини Тюрінга теж є математичною абстракцією, його не можна ототожнити з якимсь елементарним процесом, що відбувається у матеріальному середовищі. Крок машини Тюрінга визначається зміною її станів без пояснень того, як це відбувається.

Нормальні алгоритми Маркова також уточняють поняття "алгоритм", дають визначення кроку алгоритму через дві взаємопов'язані операції, що повторяються, – розпізнавання і підстановки. Операції розпізнавання і підстановки нагадують дитячі ігри в складанні кубиків. Тому ці дві операції можна вважати загальнозрозумілими. Таким чином, властивість "елементарність" у

ФАС набуває значення загальної зрозумілості, або простоти і локальності [3]. Інший підхід до тлумачення цієї властивості: обираються кінцевий набір початкових об'єктів і кінцевий набір способів побудови нових об'єктів, які виголошуються елементарними [9]. Зрозуміло, що такі тлумачення елементарності є умовними і не можуть вважатися як точні математичні визначення. Точне математичне визначення властивості "елементарність" можливе лише за умови використання ФАС з апаратним обчислювачем.

**Апаратно-програмне тлумачення алгоритму**

Теорія алгоритмів, побудована на основі абстрактного обчислювача, не уможливлює повною мірою використати досягнуті здобутки для синтезу, аналізу і оптимізації комп'ютерних засобів. Більше того, деякі висновки вступають у суперечливість з практикою комп'ютерних обчислень. Наприклад, теорема про лінійне прискорення [5]. Наближення теорії алгоритмів до потреб комп'ютерної техніки є необхідним. В цьому напрямку зроблено багато. Останнє тлумачення "алгоритм – це припис..." дає можливість під словом "припис" розуміти апаратні або апаратно-програмні засоби. Для фахівців з комп'ютерної інженерії – це аксіома.

***Аксіома**. "Алгоритм можна реалізувати апаратними або апаратно-програмними засобами, але не можна реалізувати лише програмно".*

Згадані формулювання поняття "алгоритм" і ця аксіома приводять до такого інтуїтивного тлумачення:

*"Алгоритм – це фіксована для розв'язання деякого класу задач конфігурація апаратно-програмних засобів перетворення, передавання і зберігання даних, що задає обчислювальний процес, який починається з деякої системи початкових даних (потенційно нескінченної) і скерований на отримання результату, повністю визначеного цими початковими даними".*

Це визначення відрізняється від попереднього тлумачення тим, що замість слів "точний припис" використані слова комп'ютерної термінології – "…фіксована для деякого класу задач конфігурація апаратних засобів перетворення, передавання і зберігання даних…". Останнє тлумачення не має принципових розбіжностей з існуючим тлумаченням: воно не суперечить переліку властивостей алгоритму, воно має всі визначені параметри. Разом з тим воно уможливлює побудувати апаратно-програмну модель алгоритму, розширити перелік властивостей і характеристик складності комп'ютерних засобів.

Наведене формулювання теж є інтуїтивним, математично нестрогим. Принципова відмінність цього тлумачення – розширення поняття "припису", надання йому сучасного комп'ютерного змісту. Таке тлумачення дає змогу створювати моделі алгоритму, які враховують у своєму складі апаратно-програмні засоби в явній формі. Прикладом такої моделі є SH-модель (SH-Software/Hardware) [8].

***Визначення:*** SH-модель – це сімка

$$B =< D, Q, q_0, q_f, G, P, M >, \tag{3}$$

де $D-$ кінцева множина символів зовнішнього алфавіту; $Q$ – кінцева множина станів SH-моделі; $q_0$ *і* $q_f$ – початковий і кінцевий стани, $q_0, q_f \in Q$; $G$ – конфігурація апаратних засобів моделі; $G = (X, U)$, де $X$ – множина елементарних перетворювачів; $U$ – множина міжз'єднань; $P$ – програма, $P = \left\{ y_i \middle| \, i = \overline{1, \mathrm{I}} \right\}$; $M$ – пам'ять.

SH-модель не має раз і назавжди встановленої структури. Однак кожна конкретна модель алгоритму стосовно апаратної побудови має точно окреслену структуру, яка складається з двох множин: множини елементарних перетворювачів і множини міжз'єднань:

$$\begin{aligned} E &= \{e_1, e_2 ... e_n\}; \\ U &= \{u_1, u_2 ... u_m\}. \end{aligned} \tag{4}$$

**Елементарні перетворювачі**

Апаратні засоби містять один або декілька елементарних перетворень. Елементарний *i*-тий перетворювач $x_i$ є одиницею апаратної складності.

$$\forall x_i = 1. \tag{5}$$

Елементарний перетворювач $x_i$ перетворює деяку сукупність початкових даних $d_i$ в сукупність вихідних даних $d_i^{'}$:

$$x_i : \{d_i\} \to \{d_i^{'}\}. \tag{6}$$

Часова складність $l_i$ перетворювача прийнята такою, що дорівнює одиниці:

$$\forall i, l_i = 1. \tag{7}$$

Елементарний перетворювач подається у вигляді чорної скриньки. Формальне визначення операції перетворення даних елементарним перетворювачем можливе із застосуванням будь-якої відомої моделі алгоритму, включаючи SH-модель. Звідси випливає, що поняття "SH-модель" і "елементарний перетворювач" мають ієрархічний зміст. Таким чином, для SH-моделі справедлива властивість ієрархічності. Зазначимо, що абстрактні моделі обчислювачів такої властивості не мають, що істотно обмежує їх використання для дослідження сучасних комп'ютерних засобів.

Використання елементарних перетворювачів у складі моделі алгоритму робить необхідним врахування системи зв'язків між ними $U = \{U_i \mid i = 1, j\}$. Без цієї системи зв'язків дослідження апаратної реалізації алгоритму буде неповним.

Конфігурація зв'язків впливає на інші характеристики комп'ютерних засобів, зокрема змінює значення апаратної і програмної складностей. Звідси вимога: врахування конфігурації зв'язків або структурної складності – додаткової характеристики SH-моделі.

**Властивості SH-моделі**

SH-модель має всі властивості, які випливають з вербального інтуїтивного тлумачення алгоритму.

*Дискретність.* Робота SH-моделі здійснюється множиною обмежених у часі кроків. Кожний крок може включати елементарні операції перетворення, передачу даних від одного елементарного перетворювача до іншого, а також операцію запису даних в елементи пам'яті.

*Елементарність.* Операції перетворення, передачі і запису даних в елементи пам'яті SH-моделі є простими і локальними в просторі і часі. Слову "елементарність" надається ієрархічний зміст.

*Детермінованість.* Кожний крок алгоритму повністю визначений функцією елементарного перетворювача і командою програми. Напрямок передачі даних від одного елементарного перетворювача до іншого точно визначений напрямком, що задають з'єднання або команди програми.

*Спрямованість.* Якщо процес перетворення даних не може відбуватися за конфігурацією зв'язків і програмою, то повинна бути вказівка, що робити в такій ситуації.

*Масовість.* Ця властивість має дворівневий зміст:

1) універсальність – одна і та сама SH-модель може бути застосована для розв'язання будь-якої задачі, що формалізується;

2) масовість у традиційному тлумаченні – одна і та сама SH-модель може бути застосована для будь-якої кількості задач, які відрізняються набором вхідних даних при постійному правилі безпосереднього перероблення.

*Ієрархічність.* Кожний елементарний перетворювач може бути поданий SH-моделлю нижчого ієрархічного рівня. З іншого боку, кожна SH-модель може бути використана як елементарний перетворювач вищого ієрархічного рівня.

### Параметри SH-моделі

Параметри алгоритму – правило початку, правило закінчення задаються програмою. Системи проміжних і кінцевих результатів, правило безпосереднього перероблення задаються апаратними або апаратно-програмними засобами. Система початкових даних задається зовнішніми по відношенню до SH-моделі пристроями пам'яті. Правило вводу і правило виводу задаються апаратно-програмними засобами [6].

### Характеристики складності SH-моделі

У процесах синтезу, аналізу і оптимізації SH-моделей пропонується використовувати п'ять характеристик складності: апаратну, часову, програмну, структурну і місткісну.

*Апаратна складність* – кількість елементарних перетворювачів і елементів тимчасової пам'яті деякого ієрархічного рівня апаратних засобів SH-моделі:

$$A = |X|, \tag{8}$$

де $X$ – множина елементів схеми.

Визначення відображає ієрархічну побудову комп'ютерних засобів. Якщо під SH-моделлю розуміти операційний пристрій, то елементарними перетворювачами є однорозрядні комірки або вентилі, для рівня регістрових передач елементами є операційні пристрої, на системному рівні SH-моделлю є комп'ютер тощо.

Разом з тим при оцінці апаратної складності можливий інший традиційний підхід. Апаратну складність мікропроцесорів можна визначати кількістю транзисторів, розташованих на кристалі. Постійне збільшення апаратної складності створює сприятливі умови для розширення функціональних можливостей комп'ютерів, для покращання практично всіх споживчих характеристик процесорів, зокрема точності обчислень, збільшення продуктивності (без збільшення фізичних розмірів комп'ютерів), розширення функцій.

Наведене визначення апаратної складності не суперечить поняттю "об'єм обладнання", що використовується в обчислювальній техніці.

*Часова складність*. В метричній теорії під часовою складністю розуміють кількість елементарних операцій, наприклад, кроків машини Тюрінга. В теорії software/hardware часова складність визначається дещо інакше.

Часова складність SH-моделі визначається кількістю елементів схеми, розташованих вздовж максимального критичного шляху розповсюдження сигналу:

$$L = |\max X_i|, \tag{9}$$

де $\max X_i$ – кортеж елементів SH-моделі, що належать до максимального критичного шляху розповсюдження сигналу, включаючи повторні проходження елементів у циклі.

Одиницею часової складності є елементарний перетворювач деякого ієрархічного рівня схеми. Перехід від часової складності до визначення часу спрацювання схеми наведений у формулі

$$T = \sum_{\tau_{e_i} \in \max |e_i|} \tau(e_i), \tag{10}$$

де $\tau(e_i)$ – реальна затримка сигналу на елементі $e_i$.

Апаратна і часова складності описують властивості комп'ютерних засобів, які прийнято відносити до технічних характеристик. SH-моделі, крім апаратної і часової, характеризуються додатково програмною і структурною складностями.

*Програмна складність*. Проаналізуємо часову діаграму роботи деякого пристрою на рівні регістрових передач з точки зору інформації, що міститься в ній. Часова діаграма являє собою

двовимірну таблицю, на осі абсцис якої позначені дискрети часу, на осі ординат – входи керування пристроями функціональних схем.

На процесорному рівні кожній асемблерній команді відповідає власна мікропрограма (часова діаграма). Для кожної мікропрограми конфігурація положень сигналів керування є фіксована. Використовуючи логарифмічну міру ступеня нерегулярності (ентропії) часової діаграми, запишемо

$$P = -F \log_2 \frac{F}{n \cdot m}, \tag{11}$$

де $F = \sum_L f_l$; $n$ – кількість входів керування; $m$ – кількість дискрет часу часової діаграми; $f_l$ – кількість сигналів керування $l$-того фрагмента часової діаграми для обраного рівня ієрархії побудови апаратних засобів; $L$ – кількість фрагментів часової діаграми, конфігурації яких не повторюються.

Фрагменти, що повторюються, не повинні враховуватися при розрахунку $F$ для формули (11). Це безпосередньо випливає з визначення програмної складності як ентропії часової діаграми. Умова вибору фрагментів така:

$$\forall i, j; i \neq j \left\{ \left[ \varphi_i \cap \varphi_j \neq \varphi_i \vee \varphi_j \right] \Rightarrow |f_l| = |\varphi_i| + |\varphi_j| \right\} \vee \left\{ \left[ \varphi_i \cap \varphi_j = \varphi_i \vee \varphi_j \right] \Rightarrow |f_l| = |\varphi_i| \right\}, \tag{12}$$

де $\varphi_{i,} \varphi_j$ – фрагменти часової діаграми.

*Структурна складність* відображає ступінь нерегулярності міжзв'язків схеми деякого рівня ієрархії побудови апаратних засобів. Структурна складність SH-моделі визначається аналогічно, як і програмна. Відмінність полягає лише в тому, що об'єктом розрахунків є матриця інциденцій. Структурна складність алгоритмічного пристрою – це ентропія матриці інциденцій [8]:

$$S = -E \log_2 \frac{E}{q \cdot r}, \tag{13}$$

де $E$ – кількість елементів матриці інциденцій системи; $E = \sum_L \varepsilon_l$; $q \cdot r$ – розмір матриці; $\varepsilon_l$ – кількість міжз'єднань $l$-го неповторюваного фрагмента матриці інциденцій схеми; $L$ – кількість фрагментів матриці інциденцій, які не повторюються

У (13) використовується лише множина з'єднань фрагментів матриці, що не повторюються. Вибір фрагментів проводиться з логічних умов:

$$\forall i, j; i \neq j \left\{ \left[ \psi_i \cap \psi_j \neq \psi_i \vee \psi_j \right] \Rightarrow |\varepsilon_l| = |\psi_i| + |\psi_j| \right\} \vee \left\{ \left[ \psi_i \cap \psi_j = \psi_i \vee \varphi_j \right] \Rightarrow |\varepsilon_l| = |\psi_i| \right\}, \quad i, j \in L, \tag{14}$$

де $\psi_i, \psi_j$ – і-тий та j-тий фрагменти матриці інциденції.

Структурну складність отримують в три етапи:

1. Схема SH-моделі перетворюється в орграф.
2. Орграф кодується у вигляді матриці інциденції.
3. Розраховується значення нерівномірності матриці інциденції.

*Місткісна* складність SH-моделі дорівнює кількості комірок зовнішньої пам'яті, яка потрібна для розв'язання цієї задачі.

**SH-модель – модель СМО**

SH-модель комп'ютерної системи може використовуватись для аналізу інших систем, наприклад, систем масового обслуговування (СМО). У цьому разі еквівалентом апаратної складності є кількість елементів СМО, часової – кількість кроків на шляху транзакта від початку до

закінчення, структурної – ступінь неоднорідності конфігурації моделі, програмної – ступінь нерівномірності роботи переходів, місткісна – відображає сукупність довжин черг для їх максимальних значень.

Розширення і формалізація характеристик СМО поглиблює можливості їх дослідження на етапах синтезу, аналізу і оптимізації. Для багатьох СМО зв'язком для побудови функції оптимізації може бути прийнята вартісна вага кожної характеристики. Наприклад, на сьогодні апаратна складність комп'ютерної системи значно менша від часової складності. Програмна складність істотно перевищує вартісну вагу структурної складності. Нові можливості відкриваються для оптимізації СМО некомп'ютерного застосування, наприклад, комерційних систем.

**Висновки**

1. Існуючі системи обчислень та перетворення даних потребують адекватних моделей алгоритму. Для комп'ютерних засобів пропонується використання апаратно-програмної моделі (SH-моделі), вона відрізняється від моделей з абстрактним обчислювачем наявністю апаратних засобів у визначенні.

2. Базисним елементом побудови SH-моделі є елементарний перетворювач даних. Його наявність дає змогу математично точно визначити властивість алгоритму "елементарність", додатково ввести властивість "ієрархічність" при описі моделі, уточнити поняття властивості "масовість".

3. Використання SH-моделі алгоритму уможливлює проводити синтез, аналіз, оптимізацію комп'ютерних засобів за п'ятьма характеристиками складності: апаратною, часовою, програмною, структурною, місткісною.

4. Програмна і структурна складності SH-моделі визначаються кількістю інформації, що міститься в апаратно-програмних комп'ютерних засобах.

5. Характеристики складності SH-моделі можуть бути використані для дослідження інших систем масового обслуговування.

*1. Черкаський М.В., Мітьков В.С. Історичний аспект складності алгоритму // Вісник НУ "Львівська політехніка" "Комп'ютерні системи та мережі". – Львів. – 2002. – №463. – С.111–118. 2. Трахтенброт Б.А. Алгоритмы и вычислительные автоматы. – М., 1974. 3. Марков А.А. Теория алгорифмов. – М.–Л., 1954. (Труды МИАН. Т. 42). 4. Мальцев А.И. Алгоритмы и рекурсивные функции. – М., 1986. 5. Успенский В.А., Семенов А.Л. Теория алгоритмов: основные открытия и приложения. – М., 1987. 6. Математическая энциклопедия / Гл. ред. И.М. Виноградов. Т. 1. – М., 1977. 7. Кормен Т., Лейзерсон Ч., Ривест Р. Алгоритмы: построение и анализ / Пер. с англ. – М., 2001. 8. Черкаський М.В. SH-модель алгоритму // Вісник НУ "Львівська політехніка" "Комп'ютерна інженерія та інформаційні технології".– Львів. – 2001. – №433. – С.127–134. 9. Кузнецов О.П, Адельсон-Вельский Г.М. Дискретная математика для инженера. – М., 1980. 10. Стивенс, Род. Visual Basic. Готовые алгоритмы / Пер. с англ. – М., 2000.*