УДК 004.27

О.П. КУРГАЄВ, І.В. САВЧЕНКО

СТРУКТУРА ТА ДОСЛІДЖЕННЯ ШВИДКОДІЇ СИСТЕМИ ОБРОБКИ ЗНАНЬ

Анотація. *Розробка нових, більш продуктивних, архітектур комп'ютерів для обробки знань залишається актуальною задачею. В роботі запропоновано структуру інтерпретатора баз знань та виконано емпіричне дослідження продуктивності системи обробки знань. У дослідженнях застосовано метод апроксимації. Отримані результати свідчать про доцільність апаратної реалізації засобів інтерпретації баз знань. Запропонований інтерпретатор може бути використаний при побудові сучасних систем обробки знань.*
Ключові слова: *система обробки знань, швидкодія, макетна плата, архітектура, процесор, набір команд.*

Аннотация. *Разработка новых, более производительных, архитектур компьютеров для обработки знаний остается актуальной задачей. В работе предложена структура интерпретатора баз знаний и проведено эмпирическое исследование быстродействия системы обработки знаний. В исследованиях использован метод аппроксимации. Полученные результаты свидетельствуют о целесообразности аппаратной реализации средств интерпретации баз знаний. Предложенный интерпретатор может быть использован при построении современных систем обработки знаний.*
Ключевые слова: *система обработки знаний, быстродействие, макетная плата, архитектура, процессор, набор команд.*

Abstract. *Nowadays developing of new and more productive computer architectures for knowledge processing is a vital task. The interpreter structure of intelligent databases is suggested. The empirical research of the system performance of knowledge processing is conducted. There was used an approximation method in the researches. The received results prove the feasibility of hardware implementation of intelligent databases. The proposed interpreter can be used in the construction of modern knowledge processing systems.*
Keywords: *knowledge processing system, performance, prototyping board, architecture, processor, command set.*

1. Вступ

Системи обробки знань відіграють значну роль у житті людства і з кожним роком набувають все більшого розповсюдження. Дані системи застосовуються у таких прикладних областях [1]: управління, контроль виробничих процесів, діагностика, виявлення несправностей, робототехніка, обробка зображень, машинний зір, медичні системи, моніторинг та прогнозування на фінансовому й фондовому ринках та ін. Засобами систем обробки знань створюють прикладні системи, що відповідають найбільш суттєвим ознакам концепту розумна діяльність [2–4]: розпізнавання, перетворення і розуміння образів (природномовних, зорових, тактильних тощо), вирішення задач у просторі станів, прийняття рішень, адаптація та навчання на основі накопиченого досвіду, цілеспрямована поведінка, самоорганізація систем тощо.

Об'єм та складність задач, які вирішують сучасні системи обробки знань, постійно зростають. Це змушує розробників апаратних засобів шукати шляхи створення більш швидких, надійних та одночасно економічних архітектур комп'ютерних систем. Принциповими в даному напрямі залишаються задачі [5, 6] підвищення продуктивності процесорів,

ISSN 1028-9763. Математичні машини і системи, 2012, № 2

швидкодії пам'яті, портів вводу-виводу. В роботі головну увагу зосереджено саме на задачі підвищення продуктивності роботи процесора.

Для підтримки роботи систем обробки знань сьогодні застосовуються комп'ютери з такою архітектурою [5]: багатоядерні, мультитрейдові, суперскалярні та архітектури з командним словом великої довжини. На сьогодні підвищення продуктивності процесора досягається за рахунок розв'язання таких задач [5, 6]: зменшення розмірів логічних елементів та щільності їх розташування; підвищення розміру та швидкодії кеш-пам'яті; внесення змін в архітектуру та організацію процесора (використання різних форм паралелізму, конвеєризації команд, введення додаткових наборів спеціалізованих пристроїв та команд процесора). Серед робіт, які глибоко досліджують проблему підвищення продуктивності процесора за рахунок розширення команд процесора, можна виділити роботу [7]. Прикладом використання такого підходу для підвищення продуктивності систем обробки знань є набір команд SSE4.2, який використовується у сучасних процесорах [8]: для ефективної обробки строк у XML-файлах, синтаксичного розбору речення, створення міток, оцінки регулярних виразів та пошуку вірусів.

Стрімкий розвиток технологій проектування та розробки систем на кристалі зумовив виникнення робіт, які пропонують використання проблемно-орієнтованих архітектур для розв'язку задач у вузьких предметних областях. У [9] наведено широкий огляд апаратних реалізацій систем обробки знань із використанням реконфігурованих засобів, генетичних алгоритмів, нечіткої логіки, нейронних мереж та паралельних алгоритмів. Однак семантичний розрив, недостатня гнучкість архітектури, неефективне використання пам'яті та апаратних витрат зумовлюють необхідність розробки нових наборів команд та архітектур співпроцесорів для ефективної підтримки роботи систем обробки знань [4].

Метою роботи є емпіричне дослідження швидкодії спеціалізованого інтерпретатора баз знань та порівняння експериментальних даних із даними, отриманими при використанні відомих засобів інтерпретації на основі універсальної архітектури.

2. Структура системи обробки знань

Для проведення емпіричних досліджень швидкодії було реалізовано систему обробки знань (рис. 1) на базі макетної плати M1AGL-DEV-KIT-SCS корпорації Actel. На базі кристалу програмованої логічної інтегральної схеми (ПЛІС) M1AGL600 FPGA реалізовано систему обробки знань, яка містить такі модулі: процесор CortexM1, контролер пам'яті, регістрову пам'ять, процесорну шину AHB, периферійну шину APB, міст сполучення процесорної та периферійної шини, апаратний інтерпретатор баз знань, UART-контролер, порти вводу-виводу, таймер, контролер переривань. До складу системи обробки знань також входить головний комп'ютер.

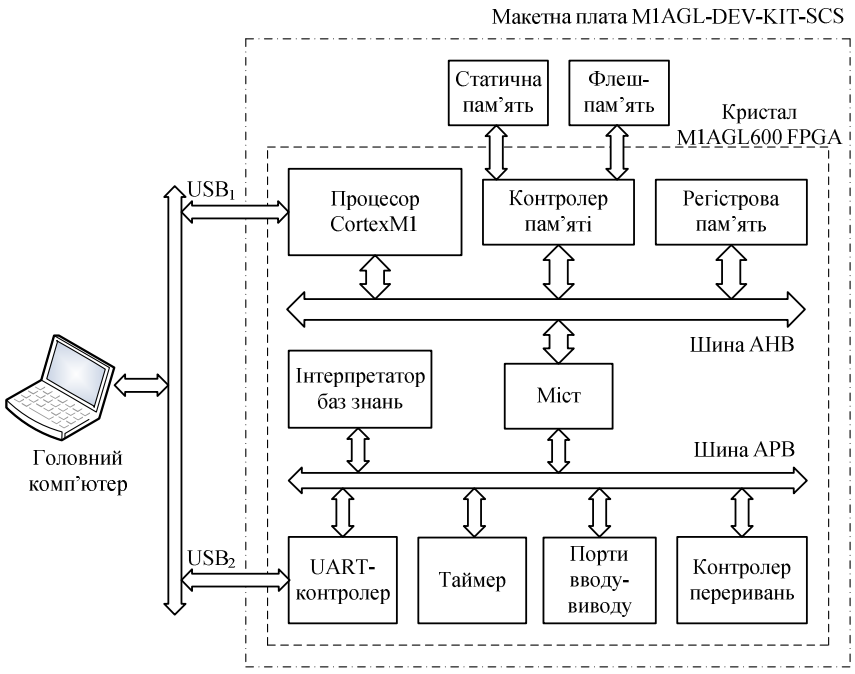

Рис. 1. Структура системи обробки знань

З метою розширення набору команд універсального процесора (Cortex M1) у систему обробки знань інтегровано додатковий модуль – інтерпретатор баз знань зі спеціальним набором команд для ефективної підтримки роботи системи обробки знань.

До складу функцій CortexM1 належать ініціалізація системи обробки знань; отримання завдання від головного комп'ютера; передача отриманого завдання до інтерпретатора баз знань; виконання термінальних програм на замовлення інтерпретатора баз знань; передача результатів виконання термінальних програм до інтерпретатора баз знань; зчитування із регістрів інтерпретатора баз знань результатів опрацювання поставленого завдання та передача отриманих результатів головному комп'ютеру.

Флеш-пам'ять є енергонезалежною й призначена для постійного збереження таких даних системи обробки знань: пам'ять програм, база знань, словник термінів. Оскільки флеш-пам'ять достатньо повільна, однак енергонезалежна, то дані в ній зберігаються лише під час відсутності живлення системи. Під час ініціалізації системи всі дані переписуються у більш швидку – статичну пам'ять.

Статична пам'ять у зв'язку з її високою швидкодією безперервно використовується у процесі роботи системи обробки знань і призначена для збереження таких даних: пам'ять програм, програмний стек Cortex-M1, база знань, словник термінів, вхідний масив, вихідний масив.

Регістрова пам'ять використовується процесором CortexM1 для збереження локальних даних (змінних, масивів, констант), швидкість доступу до яких має бути максимальною.

Інтерпретатор баз знань опрацьовує знання, представлені метамовою у формі взаємопов'язаних визначень понять деякої предметної області [4].

Контролер переривань використовується для синхронізації процесів обміну командами і даними між процесором CortexM1 та усіма функціональними модулями, розташованими на процесорній та периферійній шинах системи. Контролер переривань також відіграє суттєву роль у процесі обміну даними між головним комп'ютером і системою обробки знань.

UART-контролер виконує функцію пристрою сполучення або мосту між системою обробки знань і головним комп'ютером.

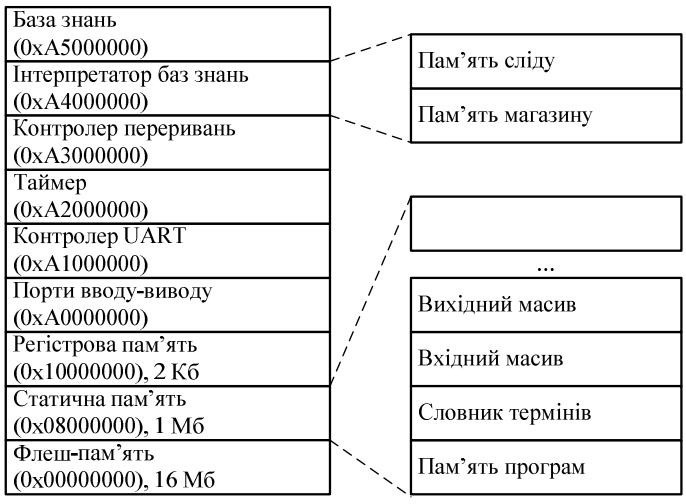

Рис. 2. Структура адресного простору системи обробки знань

Таймер та порти вводу-виводу використовуються спільно для візуального відображення поточного стану системи обробки знань на світлодіодній панелі у вигляді рухомого рядка.

Доступ процесора Cortex-M1 до ресурсів функціональних модулів (пам'ять, регістри), а також обмін командами й даними між ними, відбувається через виділений для кожного із них окремий адресний простір, структуру якого зображено на рис. 2.

Слід зазначити, що статична пам'ять (рис. 2) включає пам'ять програм, яка містить програму, що керує й підтримує роботу системи обробки знань. Дана програма забезпечує ініціалізацію системи обробки знань; отримання завдання від головного комп'ютера; передачу отриманого завдання до апаратного інтерпретатора баз знань; виконання термінальних програм на замовлення інтерпретатора баз знань; пере-

дачу результатів виконання термінальних програм до інтерпретатора баз знань; зчитування із регістрів інтерпретатора баз знань результатів опрацювання поставленого завдання та передачу отриманих результатів до головного комп'ютера.

3. Структура інтерпретатора баз знань

Інтерпретатор баз знань опрацьовує знання, представлені метамовою у формі взаємопов'язаних визначень понять деякої предметної області. Кожне із понять визначається як альтернатива або послідовність деяких понять (серед яких може бути і поняття, що визначається), кожне із яких може бути ітерацією деякої структури визначень або задано визначеннями. Будь-яке просте поняття (найнижчого рівня складності) визначається у формі константи або процедури (термінальної програми) [4].

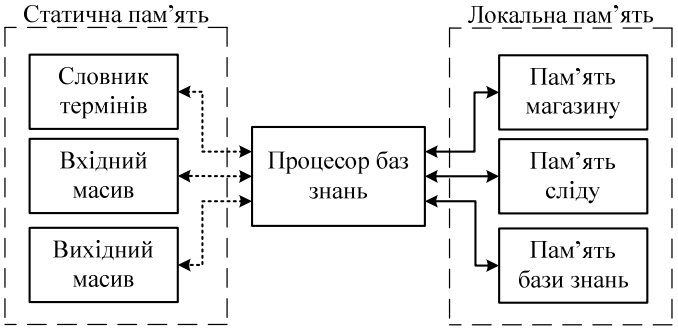

Рис. 3. Структура інтерпретатора баз знань

Інтерпретатор баз знань містить такі основні компоненти (рис. 3): процесор баз знань, локальну та статичну пам'ять. Процесор баз знань складається із пристрою управління, реалізованого у вигляді цифрового синхронного автомата Мура, та пристрою обробки даних із набором регістрів, в яких зберігається поточний стан інтерпретатора. Статична пам'ять, розміщена поза кристалом ПЛІС, містить такі пам'яті інтерпретатора баз знань: словник термінів, вхідний масив та вихідний масив. Процесор баз знань не має прямого доступу до даних статичної пам'яті, а отримує лише логічні результати їх опрацювання процесором Cortex-M1. Локальна пам'ять, яка розміщена на кристалі ПЛІС, містить такі пам'яті: бази знань, магазину та сліду.

Словник термінів містить множину записів, кожний із яких складається із двох полів: «Ім'я поняття» та «Адреса структури визначення». Поле «Ім'я поняття» містить послідовність алфавітно-цифрових символів і використовується для розіменування імені шуканої структури у фізичну адресу її розміщення у базі знань. Фізична адреса даної структури міститься у полі «Адреса структури визначення».

Вхідний та вихідний масиви у формі ASCII-коду містять відповідно: суб'єкт категоричного висловлювання, предикат якого потрібно довести, та суб'єкт, отриманий внаслідок доведення.

Пам'ять магазину містить поточний стан інтерпретації баз знань у формі множини записів фіксованої довжини. Кожен запис містить поля: координати вхідного та вихідного масивів, координати сліду, адресу поточного слова фрейму, кількість успішно виконаних ітерацій, режим інтерпретації.

Пам'ять сліду містить масив записів фіксованої довжини, який відображає хід інтерпретації понять у формі дерева виводу. Кожен запис може містити дані двох типів: фізичну адресу успішно інтерпретованої альтернативи або кількість успішно виконаних ітерацій.

Пам'ять бази знань містить інформаційну структуру знань, яка складається із множини фреймів. Кожен фрейм бази знань містить голову та елементи, пов'язані одним відношенням: кон'юнкції, диз'юнкції або ітерації. Голова та будь-який з елементів займають декілька послідовно розташованих комірок пам'яті. Елементи одного масиву, пов'язані відношенням диз'юнкції або кон'юнкції, розташовані у наступних сусідніх комірках пам'яті. Голова фрейму розташована першою в даній послідовності, при цьому розмір фрейму (число елементів в одному фреймі) може бути довільним. Фрейм ітерації складається із двох

компонент: голови та ітерованого елемента. Ітерований елемент представляється посиланням на деякий фрейм, який є головною компонентою опису даного елемента. Цим забезпечується зв'язність різних фреймів в єдину багаторазово вкладену структуру, що допускає рекурсивні конструкції. Складність опису (потужність множини, кількість масивів) може бути довільною і обмежується ресурсами конкретної реалізації.

Ті поняття бази знань, які не мають опису через інші поняття, є термінальними і визначаються посиланням на термінальні програми, що в подальшому виконуються універсальним процесором (CortexM1). Термінальні програми реалізовано у вигляді скомпільованого коду процедур та функцій.

Формально базу знань *KB* можна визначити такою четвіркою [4]:

$$KB =< A_{TR},\ A_{CON}, S, P >, \tag{1}$$

де A_{TR} – множина термінальних програм, які визначено поза базою знань і які виконуються на універсальному процесорі; A_{CON} – множина імен понять та текстових констант; S – поняття найвищого рівня складності, яке визначається базою знань; P – інформаційна структура поняття S.

Детальний опис алгоритму інтерпретації наведено у [4]. Серед задач, які виконуються у процесі інтерпретації знань, можна виділити задачі аналізу та породження предметних значень понять. Аналіз зводиться до доведення простого категоричного висловлення, предикат якого – поняття бази знань, а суб'єкт – дані із вхідного масиву. Логічний результат виконання задачі аналізу – істина або хибність. Задача породження є оберненою до задачі аналізу і зводиться до запису у вихідний масив значення суб'єкта, який отримано внаслідок доведення останнього судження, предикат якого – відповідне поняття. Логічний результат виконання операції породження – завжди істина.

Знання у базі знань задають статичну форму знань. Головний її компонент – інформаційна структура P, яка є системою правил виводу загального судження. Даний статичний компонент доповнюється динамічним компонентом (процедурою інтерпретації), яка розкриває зміст інформаційної структури та використовує її в процесі виводу судження.

4. Проведення емпіричного дослідження швидкодії системи обробки знань

У процесі експериментального дослідження враховано такі фактори:

1) Як показник оцінки швидкодії слід обрати величину, що характеризує інтервал часу між початком запуску роботи інтерпретатора до моменту переходу його у стан очікування нового завдання.

2) У процесі проведення емпіричного дослідження виконується порівняння ефективності процесів інтерпретації на реалізованому апаратно інтерпретаторі баз знань (АІБЗ) та на реалізованому програмно інтерпретаторі баз знань (ПІБЗ). АІБЗ реалізовано на макетній платі M1AGL-DEV-KIT-SCS. ПІБЗ скомпільовано в середовищі Microsoft Visual Studio 2010 із застосуванням бібліотеки Boost/Spirit [10] та реалізовано у вигляді ПІБЗ2 та ПІБЗ4. ПІБЗ2 функціонує на базі персонального комп'ютера з такими характеристиками: операційна система – MS Windows XP Professional 32-bit SP3; універсальний процесор – Intel Mobile Core 2 Duo T8100 2.10 ГГц; кількість процесорних ядер – 2; розмір оперативної пам'яті – 2 Гб. ПІБЗ4 функціонує на базі персонального комп'ютера з такими характеристиками: операційна система – Windows Server 2008 R2 Standard SP 1; універсальний процесор – Intel Xeon CPU E5504 2.00 ГГц; кількість процесорних ядер – 4; розмір оперативної пам'яті – 12 Гб.

3) Вимірювання часу виконання поставленого завдання у секундах не цілком коректне, оскільки центральні процесори АІБЗ та ПІБЗ виготовлені за різними технологіями і використовують різні тактові частоти (16 МГц – АІБЗ; 2.1 ГГц – ПІБЗ). Найбільш коректно-

ний результат порівняння може бути отриманий лише при однакових тактових частотах обох процесорів. Для того, щоб позбутись впливу технологічних особливостей реалізації, доцільно порівнювати не абсолютний час виконання (в секундах), а число тактових імпульсів процесора, витрачених на виконання процесу інтерпретації.

4) Найбільш точна оцінка вимірювання тривалості виконання процесу інтерпретації для ПІБЗ досягається зчитуванням даних із процесорного регістра RDTSC – 64-розрядного лічильника тактів процесора. Цей метод вимірювання тривалості виконання деякої функції є найточнішим. Основною перевагою даного методу є те, що постійна зміна тактової частоти процесора (для економії енергоспоживання) не впливає на точність вимірювання.

5) При вимірюванні числа процесорних тактів ПІБЗ виникає ускладнення, пов'язане з тим, що сучасні обчислювальні системи є багатозадачними, тому весь процесорний час розподіляється між різними процесами. Це призводить до того, що може бути різною тривалість виконання одного й того ж завдання ПІБЗ на різних часових інтервалах. Щоб позбутись цього недоліку, тривалість виконання кожного процесу інтерпретації вимірюється декілька разів, із розподілом у часі. Для отримання оцінки середньої тривалості виконання завдання обчислюються середнє значення $\bar{x}$ та стандартне відхилення s відповідно за формулами (2) та (3):

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i , \tag{2}$$

$$s = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}\left(x_i - \bar{x_i}\right)}, \tag{3}$$

де x_i – i-тий результат вимірювання, n – кількість проведених експериментів, $\bar{x}$ – середнє арифметичне результату вимірювання, s – стандартне відхилення.

6) Найбільш точна оцінка числа процесорних тактів АІБЗ визначається за рахунок введення у нього 32-розрядного лічильника тактів процесора. Даний підхід дозволяє визначити тривалість виконання поставленого завдання із точністю до одного процесорного такту.

7) З метою порівняння ефективності алгоритму інтерпретації АІБЗ та ПІБЗ необхідно визначити закон (функцію апроксимації), який описує залежність тривалості виконання поставленого завдання від складності самого завдання. Для визначення функції апроксимації було використано інструментальний засіб Microsoft Office Excel, що входить до складу Microsoft Office. За допомогою даного інструментального засобу та на основі даних, отриманих у результаті проведення експерименту, визначено функцію апроксимації за чотирма законами: лінійний, логарифмічний, степеневий та експоненціальний. Для кожної функції апроксимації визначена величина достовірності апроксимації R^2, що розраховується за формулами 4–6 [11]:

$$R^2 = 1 - \frac{SSE}{SST}, \tag{4}$$

$$SSE = \sum_{i=1}^{n}\left(x_i - \bar{x}\right)^2 , \tag{5}$$

$$SST = \left(\sum_{i=1}^{n} x_i^2\right) - \frac{\left(\sum_{i=1}^{n} x_i\right)^2}{n}. \tag{6}$$

Серед отриманих функцій апроксимації обираємо ту, яка має найбільшу величину достовірності апроксимації.

5. Результати емпіричного дослідження

У процесі проведення експерименту було використано базу знань, що містить п'ять понять: Regular, Integer, RealValue, Identifier, Sentence. Базу знань було реалізовано у відповідності до вимог синтаксису інтерпретаторів та завантажено у ПІБЗ2, ПІБЗ4 та АІБЗ. У процесі проведення експерименту одну частину експериментальних даних було згенеровано автоматично, іншу частину – отримано із літературних джерел. Усі отримані дані було по черзі проаналізовано на ПІБЗ2, ПІБЗ4, АІБЗ та записано у відповідні таблиці для аналізу. В результаті аналізу отриманих експериментальних даних встановлено, що алгоритмічна складність алгоритму інтерпретації є лінійною. Найголовніші результати емпіричних досліджень зображено на рис. 4–8.

На рис. 4 зображено результати дослідження швидкодії роботи ПІБЗ та АІБЗ при інтерпретації поняття «Regular». Дані результати відображають залежність тривалості виконання інтерпретації поняття від довжини виразу, що інтерпретується. У результаті розрахунків отримано функції апроксимацій для ПІБЗ2, ПІБЗ4 та АІБЗ, які відповідно дорівнюють $y_{ПІБЗ2}(x) = 23{,}993x + 2183{,}5$, $y_{ПІБЗ4}(x) = 11{,}622x + 1204{,}3$ та $y_{АІБЗ}(x) = 9x + 45$. На основі отриманих даних можна зробити висновок, що алгоритмічна складність задачі «Regular» є лінійною, тобто $y_{ПІБЗ2}(n) = O(n)$, $y_{ПІБЗ4}(n) = O(n)$, $y_{АІБЗ}(n) = O(n)$. При цьому, у порівнянні з $y_{ПІБЗ2}(x)$ та $y_{ПІБЗ4}(x)$, швидкість зростання функції $y_{АІБЗ}(x)$ є найменшою (9 < 11,622 < 23,993), що доводить ефективність реалізації АІБЗ. Середня тривалість інтерпретації поняття «Regular» на ПІЗ2, ПІЗ4 та АІБЗ відповідно складає 2723,35; 1465,783 та 247,5 процесорних тактів. Виходячи із цих даних, можна зробити висновок, що середній виграш в ефективності алгоритму роботи АІБЗ, у порівнянні із ПІБЗ2 та ПІБЗ4, при обробці поняття «Regular» відповідно складає 11,0 та 5, 92 рази.

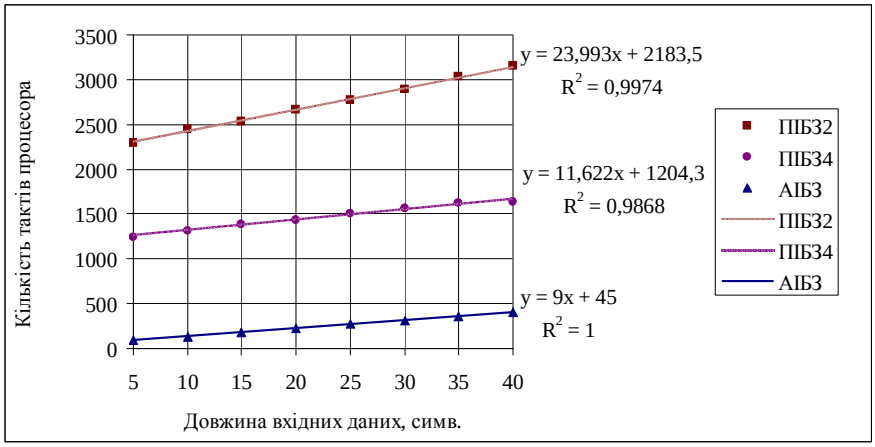

Рис. 4. Дослідження швидкодії інтерпретації поняття «Regular»

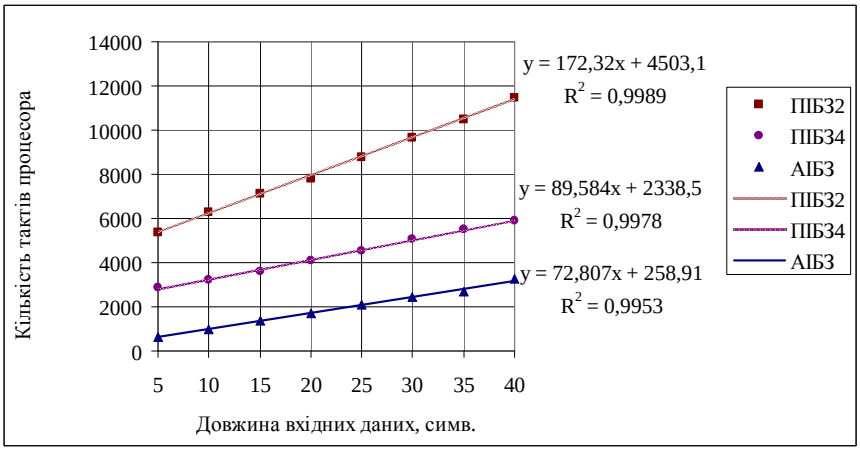

Рис. 5. Дослідження швидкодії інтерпретації поняття «Integer»

На рис. 5 зображено результати дослідження швидкодії роботи ПІБЗ та АІБЗ при інтерпретації поняття

«Integer». Дані результати відображають залежність тривалості виконання інтерпретації поняття від довжини виразу, що інтерпретується. У результаті розрахунків отримано функції апроксимацій для ПІБ32, ПІБ34 та АІБ3, які відповідно дорівнюють $y_{ПІБ32}(x)=172{,}32x+4503{,}1$, $y_{ПІБ34}(x)=89{,}584x+2338{,}5$ та $y_{АІБ3}(x)=72{,}807x+258{,}91$. Із отриманих даних слідує, що алгоритмічна складність задачі «Integer» є лінійною, тобто $y_{ПІБ32}(n)=O(n)$, $y_{ПІБ34}(n)=O(n)$, $y_{АІБ3}(n)=O(n)$. При цьому, у порівнянні із $y_{ПІБ32}(x)$ та $y_{ПІБ34}(x)$, швидкість зростання функції $y_{АІБ3}(x)$ є найменшою (72,807 < 89,584 < 172,32), що доводить ефективність реалізації АІБ3. Середня тривалість інтерпретації поняття «Integer» на ПІ32, ПІ34 та АІБ3 відповідно складає 8380,242; 4354,167 та 1897,075 процесорних тактів. Виходячи із цих даних, можна зробити висновок, що середній виграш в ефективності алгоритму роботи АІБ3, у порівнянні із ПІБ32 та ПІБ34, при обробці поняття «Integer» відповідно складає 4,42 та 2,3 рази.

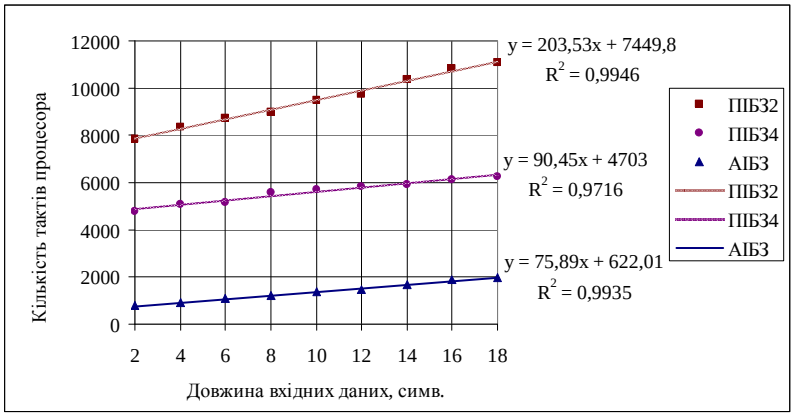

Рис. 6. Дослідження швидкодії інтерпретації поняття «RealValue»

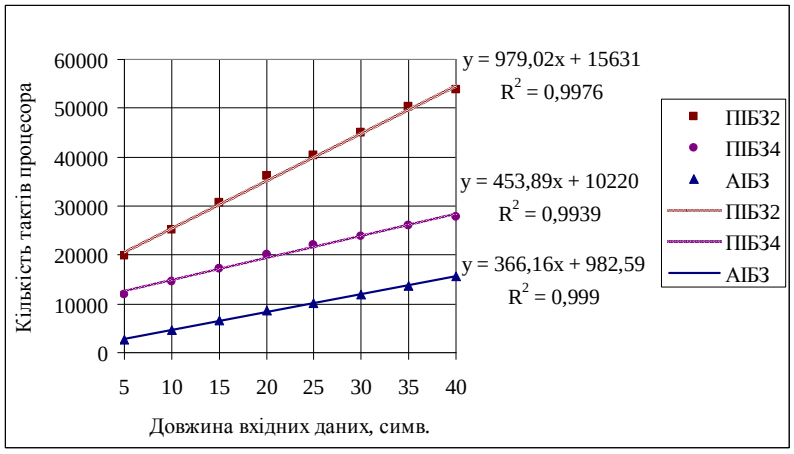

Рис. 7. Дослідження швидкодії інтерпретації поняття «Identifier»

На рис. 6 зображено результати дослідження швидкодії роботи ПІБ3 та АІБ3 при інтерпретації поняття «RealValue». Дані результати відображають залежність тривалості виконання інтерпретації поняття від довжини виразу, що інтерпретується. У результаті розрахунків отримано функції апроксимацій для ПІБ32, ПІБ34 та АІБ3, які відповідно дорівнюють $y_{ПІБ32}(x)=203{,}53x+7449{,}8$, $y_{ПІБ34}(x)=90{,}45x+4703$ та $y_{АІБ3}(x)=75{,}89x+622{,}01$. На основі отриманих даних можна зробити висновок, що алгоритмічна складність задачі «RealValue» є лінійною, тобто $y_{ПІБ32}(n)=O(n)$, $y_{ПІБ34}(n)=O(n)$, $y_{АІБ3}(n)=O(n)$. При цьому, у порівнянні із $y_{ПІБ32}(x)$ та $y_{ПІБ34}(x)$, швидкість зростання функції $y_{АІБ3}(x)$ є найменшою (75,89 < 90,45 < 203,53), що доводить ефективність реалізації АІБ3. Середня тривалість інтерпретації поняття «RealValue» на ПІ32, ПІ34 та АІ3 відповідно складає 9485,119; 5607,541 та 1380,911 процесорних тактів. Виходячи із цих даних, можна зробити висновок, що середній виграш в ефективності алгоритму роботи АІБ3, у порівнянні із ПІБ32 та ПІБ34, при обробці поняття «RealValue» відповідно складає 6,87 та 4,06 рази.

На рис. 7 зображено результати дослідження швидкодії роботи ПІБ3 та АІБ3 при інтерпретації поняття «Identifier». Дані результати відображають залежність тривалості ви-

конання інтерпретації поняття від довжини виразу, що інтерпретується. У результаті розрахунків отримано функції апроксимацій для ПІБ32, ПІБ34 та АІБ3, які відповідно дорівнюють $y_{ПІБ32}(x)=979{,}02x+15631$, $y_{ПІБ34}(x)=453{,}89x+10220$ та $y_{АІБ3}(x)=366{,}16x+982{,}59$. На основі отриманих даних можна зробити висновок, що алгоритмічна складність задачі «Identifier» є лінійною, тобто $y_{ПІБ32}(n)=O(n)$, $y_{ПІБ34}(n)=O(n)$, $y_{АІБ3}(n)=O(n)$. При цьому, у порівнянні із $y_{ПІБ32}(x)$ та $y_{ПІБ34}(x)$, швидкість зростання функції $y_{\text{АІБ3}}(x)$ є найменшою (366,16 < 453,89 < 979,02), що доводить ефективність реалізації АІБ3. Середня тривалість інтерпретації поняття «Identifier» на ПІ32, ПІ34 та АІ3 відповідно складає 37658,775; 20483,15 та 9221,15 процесорних тактів. Виходячи із цих даних, можна зробити висновок, що середній виграш в ефективності алгоритму роботи АІБ3, у порівнянні із ПІБ32 та ПІБ34, при обробці поняття «Identifier» відповідно складає 4,08 та 2,22 рази.

На рис. 8 зображено результати дослідження швидкодії роботи ПІБ3 та АІБ3 при інтерпретації поняття «Sentence». Середня тривалість інтерпретації поняття «Sentence» на ПІ32, ПІ34 та АІ3 відповідно складає 27510,556; 15284,556 та 2272,2 процесорних тактів. Виходячи із цих даних, можна зробити висновок, що середній виграш в ефективності алгоритму роботи АІБ3, у порівнянні із ПІБ32 та ПІБ34, при обробці поняття «Sentence» відповідно складає 12,11 та 6,73 рази.

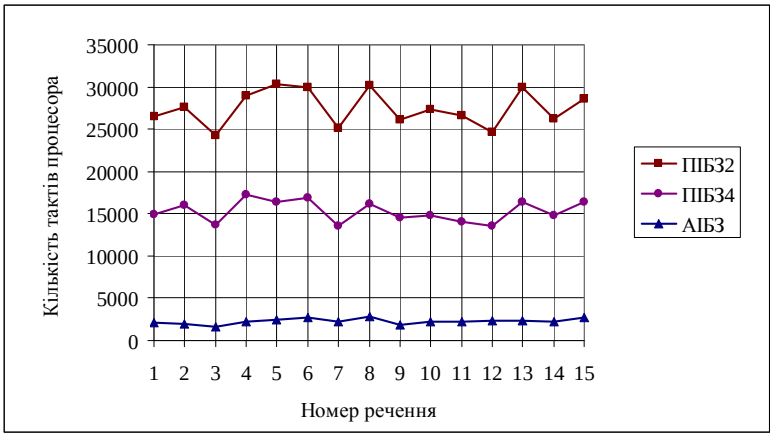

Рис. 8. Дослідження швидкодії інтерпретації поняття «Sentence»

6. Висновки

Результати емпіричних досліджень інтерпретації знань свідчать про те, що архітектура сучасних апаратних засобів не достатньо ефективна для підтримки обробки знань. Це пов'язано із наявністю семантичного розриву між поняттями відношень та їх об'єктів у мовах (моделях) представлення знань та поняттями операцій і даних, що визначаються архітектурою сучасного комп'ютера.

Для підвищення продуктивності роботи сучасних систем обробки знань доцільна розробка нових архітектур апаратних засобів, більш адекватних архітектурам систем обробки знань. Зокрема, це стосується розробки спеціалізованих апаратних засобів, які можуть розширювати набір команд універсального процесора. Перспективність даного напряму обумовлюють сучасні технології виготовлення систем на кристалі, які дозволяють швидко та за невисоку вартість реалізувати нові архітектури й поєднати їх із існуючими системами.

На основі отриманих емпіричних даних встановлено, що апаратні засоби інтерпретації знань забезпечують суттєвий виграш порівняно із програмними інтерпретаторами знань, які функціонують на базі процесорів універсальної архітектури. Середній виграш у швидкодії становить від 2,22 до 12,11 разів.

СПИСОК ЛІТЕРАТУРИ

1. Knowledge-Based Intelligent Information and Engineering Systems (KES). Aim and Scope: портал [Електронний ресурс]. – Режим доступу: http://www.kesinternational.org/aim.php.

2. Luger G.F. Artificial Intelligence. Structures and Strategies for Complex Problem Solving / Luger G.F.; Fifth ed. – Harlow: Copyright © Pearson Education Limited, 2005. – 903 p.
3. Рассел С. Искусственный интеллект: современный подход / С. Рассел, П. Норвиг; пер. с англ. – [2-е изд.]. – М.: Издательский дом «Вильямс», 2006. – 1408 с.
4. Кургаев А.Ф. Проблемная ориентация архитектуры компьютерных систем / Кургаев А.Ф. – Киев: Сталь, 2008. – 540 с.
5. Stallings W. Computer Organization and Architecture. Designing for performance / Stallings W.; Eight ed. – New Jersey: Copyright © by Pearson Education, Inc., 2010. – 763 p.
6. Hennessy J.L. Computer Architecture. A Quantitative Approach / J.L. Hennessy, D.A. Patterson; Fourth ed. – San Francisco: Copyright © by Elsevier, Inc., 2007. – 704 p.
7. Galuzzi C. The Instruction-Set Extension Problem: A Survey / C. Galuzzi, K. Bertels // Proc. of the 4th International Workshop, (London, UK, March 26–28 2008). – London, UK, 2008. – P. 209 – 220.
8. Intel 64 and IA-32 Architectures. Software Developer's Manual. Vol. 1: Basic Architecture [Електронний ресурс]. – Режим доступу: http://www.intel.com.
9. Teodorescu H.N. Hardware Implementation of Intelligent systems / Teodorescu H.N., Jain L.C., Kandel A. – Heidelberg: Copyright © Physica-Verlag, 2010. – 282 p.
10. Boost Spirit About [Електронний ресурс]. – Режим доступу: http://boost-spirit.com/home/about-2.
11. Добавление, изменение и удаление линии тренда на диаграмме [Электронный ресурс]. – Режим доступа: http://office.microsoft.com/ru-ru/excel-help/HP010342158.aspxhttp://iproc.ru/programming/windows-timers.

Стаття надійшла до редакції 02.02.2012