УДК 681.3.06

МЕТОДИ ОРГАНІЗАЦІЇ ПАРАЛЕЛЬНИХ ОБЧИСЛЕНЬ ДЛЯ БАГАТОРІВНЕВИХ НЕЧІТКИХ СИСТЕМ ТАКАГІ – СУГЕНО

С.В. Єршов, Р.М. Пономаренко

Розглянуті ярусно-паралельна та динамічна моделі для здійснення нечіткого логічного виведення в експертно-діагностичних програмних системах, бази знань яких ґрунтуються на нечітких правилах. Розроблена ярусно-паралельна та динамічна процедури нечіткого виведення, які дозволяють прискорити виконання обчислень в програмній системі, що призначена для оцінки якості наукових робіт. Побудовані оцінки ефективності ярусно-паралельної та динамічної схем обчислень за наявності складних графів залежностей між блоками нечітких правил Такагі – Сугено. Проведено порівняльну характеристику ефективності ярусно-паралельної та динамічної моделей.
Ключові слова: нечіткі системи Такагі – Сугено, ярусно-паралельна схема, динамічна схема обчислень, прискорення, граф залежностей, нечітке логічне виведення.

Рассмотрены ярусно-параллельная и динамическая модели для осуществления нечеткого логического вывода в экспертно-диагностических программных системах, базы знаний которых основываются на нечетких правилах. Разработана ярусно-параллельная и динамическая процедуры нечеткого вывода, которые позволяют ускорить выполнение вычислений в программной системе, предназначенной для оценки качества научных работ. Построены оценки эффективности ярусно-параллельной и динамической схем вычислений при наличии сложных графов зависимостей между блоками нечетких правил Такаги – Сугено. Проведена сравнительная характеристика эффективности ярусно-параллельной и динамической моделей.
Ключевые слова: нечеткие системы Такаги – Сугено, ярусно-параллельная схема, динамическая схема вычислений, ускорение, граф зависимостей, нечеткий логический вывод.

Parallel tiered and dynamic models of the fuzzy inference in expert-diagnostic software systems are considered, which knowledge bases are based on fuzzy rules. Tiered parallel and dynamic fuzzy inference procedures are developed that allow speed up of computations in the software system for evaluating the quality of scientific papers. Evaluations of the effectiveness of parallel tiered and dynamic schemes of computations are constructed with complex dependency graph between blocks of fuzzy Takagi – Sugeno rules. Comparative characteristic of the efficacy of parallel-stacked and dynamic models is carried out.
Key words: fuzzy Takagi – Sugeno systems, tiered parallel scheme, dynamic scheme of computing, speedup, dependency graph, fuzzy inference.

Останнім часом велика увага приділяється розвитку методів алгоритмів логічного виведення для вирішення наукових та практичних задач [1–3]. При застосуванні алгоритмів нечіткого логічного виведення досвід або знання експерта, що описується як база нечітких правил, може бути безпосередньо вбудований в систему. Створення методів систематичного проектування систем нечіткої логіки висвітлено в [2, 3]. В багатьох роботах акцент робиться на розбиття вхідного простору задачі з метою визначення нечітких правил і параметрів, що залучаються в нечіткі правила для однієї окремої нечіткої системи [2]. Як відомо, проблема великої розмірності нечітких систем, що можуть бути побудовані на основі нечітких правил є невирішена [3].

Для зменшення складності нечітких систем логічного виведення було запропоновано поняття нечітких багаторівневих систем, в яких входи та виходи яких пов'язані між собою [2, 3]. Побудова багаторівневої нечіткої системи є складною задачею оскільки необхідно визначити архітектуру системи (модулі, вхідні змінні кожного модуля, і взаємодія між модулями), а також правила для кожного модуля. При цьому внаслідок збільшення кількості модулів (систем нечіткого виведення) значно зростає час обчислень. Перспективним є застосування паралельних обчислень для скорочення часу обчислень при здійсненні нечіткого логічного виведення при побудові діагностичних та експертних систем, що ґрунтуються на методах нечіткої математики. Багаторівнева архітектура використовується при побудові нечітких мультиагентних систем, що функціонують в розподіленому середовищі [4–6].

Метою даної роботи є дослідження та обґрунтування методів організації паралельних обчислень для здійснення логічного виведення у нечітких багаторівневих системах Такагі – Сугено.

Системи нечіткого виведення складаються з набору правил *якщо-то*. Нечітка модель Такагі – Сугено представлена нечіткими правилами [2]:

$$R_j : \textit{якщо } x_1 \in A_{1j} \textit{ та } x_2 \in A_{2j} \textit{ та } \ldots \textit{ та } x_n \in A_{nj},$$

$$\textit{то } y = g_j(x_1, x_2, \ldots, x_n), \quad j = 1,2,\ldots,N,$$

де g_j – чітка функція від x_i. Як правило, $g_j(x_1, x_2, \ldots, x_n) = \omega_0 + \omega_1 x_1 + \omega_2 x_2 + \ldots + \omega_n x_n$. Загальний вихід нечіткої моделі отримуємо наступним чином:

$$y = \sum_{j=1}^{N} g_i \underset{i=1}{\overset{m_j}{T}} \mu_{ij}(x_i) \Big/ \sum_{j=1}^{N} \cdot \underset{i=1}{\overset{m_j}{T}} \mu_{ij}(x_i),$$

де $1 \le m_j \le n$ – це число вхідних змінних, які з'являються в антецеденті правила j, N – кількість нечітких правил, n – кількість входів, функції належності нечітких множин, T являє собою Т-норму, що використовується як операція кон'юнкції.

Нечітка система Такагі – Сугено – це одноступінчаста (елементарна) нечітка система. Багаторівневі (ієрархічні) системи нечіткого виведення не тільки забезпечують більш складну і гнучку архітектуру для моделювання систем з нелінійними залежностями, але також дозволяють значно зменшити розмір бази правил, що використовується в процесі нечіткого виведення. На рис. 1 показано приклад багаторівневої системи Такагі – Сугено, що містять 4 вхідних змінні та 2 ієрархічних рівня.

Результат роботи ієрархічної системи нечіткого виведення Такагі – Сугено може бути розрахований на пошаровій основі. ДаліНижче показаний процес обчислення для системи 1 на рис. 1. Припустимо, що значення кожної вхідної змінної (x_1, x_2, x_3, x_4) представлені двома нечіткими множинами, що задані нечіткими функціями належності $\mu(a,b,x) = 1/(1+((x-a)/b)^2)$. По-перше, обчислюється вихідне значення нечіткої підсистеми Такагі – Сугено (вершини 2).

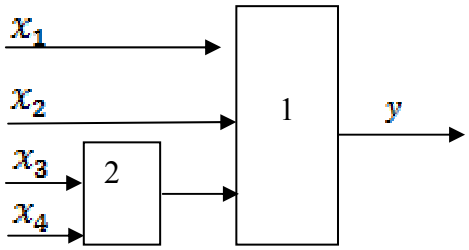

Рис. 1. Приклад багаторівневої нечітко-логічної моделі, кількість входів – 4, кількість шарів – 2

Припустимо, що використовувані нечіткі множини змінних x_3 і x_4 є A_{11}, A_{12} і A_{21}, A_{22}, відповідно. Припустимо, що параметри в консеквентах бази правил вершини 2 є c_{ij}^0, c_{ij}^1, c_{ij}^2 (i = 1,2 та j = 1, 2). Таким чином, відповідні нечіткі правила вузла 2 можна описати так:

$$R_{ij}: \text{якщо } x_3 \in A_{1i} \text{ та } x_2 \in A_{2j} \text{ то } y_{ij} = c_{ij}^0 + c_{ij}^1 x_3 + c_{ij}^2 x_4, \quad i = 1,2,\ j = 1,2.$$

Вихідне значення вузла 2 може бути розраховано на основі нечіткої моделі Такагі – Сугено наступним чином:

$$y = \sum_{i=1}^{2}\sum_{j=1}^{2} \sigma_{ij} y_{ij} \Big/ \sum_{i=1}^{2}\sum_{j=1}^{2} \sigma_{ij} \ ,$$

де $\sigma_{ij}(x_3, x_4) = \mu_{A_{1i}}(x_3)\mu_{A_{2j}}(x_4)$, i=1,2 та j=1,2.

По-друге, обчислюється загальне вихідне значення ієрархічної нечіткої моделі Такагі – Сугено. Воно обчислюється на основі трьох вхідних значень, x_1, x_2 і y, вихідного значення нечіткої підсистеми Такагі – Сугено (вузол 2).

Розглянемо організацію ярусно-паралельних обчислення при побудові систем оцінки якості наукових робіт, що подаються на конференцію [7]. Оцінка робіт здійснюється трьома експертами. Значення частинних (базових) так і укрупнених (виведених як наслідок) оцінок задаються лінгвістичними термами, наприклад, ОДНОЗНАЧНО ТАК, МАБУТЬ НІ, ВАЖКО ВИЗНАЧИТИ. На основі укрупнених оцінок трьох експертів виводиться інтегрована оцінка рівня роботи, задана числом в інтервалі [0,1], на основі якого приймається рішення (ПРИЙНЯТИ, ВІДХИЛИТИ, ДОРОБИТИ). При використанні систем Такага – Сугено робота кожного експерта може бути представлена блоками правил, кількість яких значно менша ніж при використанні єдиного блоку правил, що описують залежність прийнятого рішення стосовно рівня роботи від елементарних показників якості.

Ієрархічний зв'язок інтегральної оцінки рівня наукової роботи з оцінками за окремими показниками якості роботи представимо графом (деревом) нечіткої багаторівневої системи (рис. 2), яке визначає структуру моделі нечіткого дерева.

Вершини дерева інтерпретуються таким чином: корінь дерева – показник, що діагностується; термінальні вершини – частинні оцінки роботи; нетермінальні вершини (подвійні лінії) – згортка частинних оцінок в укрупнені. Дуги, що виходять з нетермінальних вершин дерева, відповідають укрупненим оцінкам наукової роботи.

Змістовна інтерпретація частинних та укрупнених оцінок робіт наведена в таблиці.

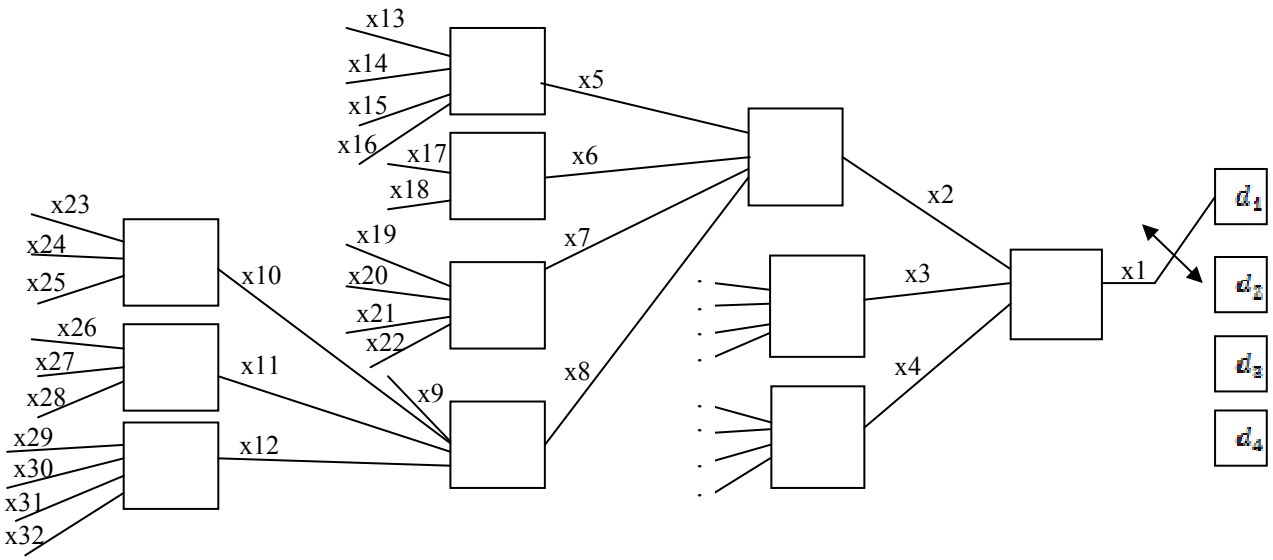

Рис. 2. Дерево логічного виведення для оцінювання рівня наукових робіт

Таблиця. Змістовна інтерпретація частинних та укрупнених оцінок наукових робіт

Назва змінної	Змістовна інтерпретація
x1	Інтегрована оцінка рівня роботи
x2	Інтегрована оцінка першого рецензента
x3	Інтегрована оцінка другого рецензента
x4	Інтегрована оцінка третього рецензента
x5	Оригінальність роботи
x6	Значення роботи для певної області досліджень
x7	Якість способу подання результатів
x8	Якість вмісту роботи
x9	Заголовок. Чи заголовок ясно відображає зміст статті?
x10	Якість анотації
x11	Якість вступу
x12	Якість представлення результатів
x13	Чи є проблема, що висвітлюється в роботі новою?
x14	Чи зазначені відмінності від існуючих підходів?
x15	Чи представлено в роботі інноваційне поєднання різних напрямків досліджень?
x16	Чи наявні в роботі перспективні ідеї?
x17	Чи вносить робота вагомий внесок в певну область досліджень?
x18	Чи стимулює робота обговорення важливих питань, або альтернативних підходів?
x19	Наявність логічної структури
x20	Чіткість викладення матеріалу
x21	Правильність з граматичної точки зору
x22	Використання адекватної термінології
x23	Інформативність анотації
x24	Відображення основних результатів в анотації
x25	Відображення перспективності подальших досліджень в анотації
x26	Правильність виділення існуючих проблем
x27	Постановка цілей дослідження
x28	Представлення наукової новизни та практичного значення у вступі
x29	Чіткість викладення основних результатів
x30	Наявність відповідних посилань на роботи в даній області досліджень
x31	Однозначність тлумачення представлених результатів
x32	Відсутність необгрунтованих припущень та домислів у результатах

Результати нечіткого логічного виведення можна контролювати за допомогою графічного інтерфейсу, фрагмент якого представлено на рис. 3. При цьому кожному з трьох експертів необхідно ввести лінгвістичні значення частинних оцінок роботи (змінні x13-x32). Система виводить рекомендовані значення оцінок експертів та показує, які правила мали найбільший вплив на висновок. Остаточне рішення приймається враховуючи відповідні рішення експертів за системою правил, яка виробляє значення x1.

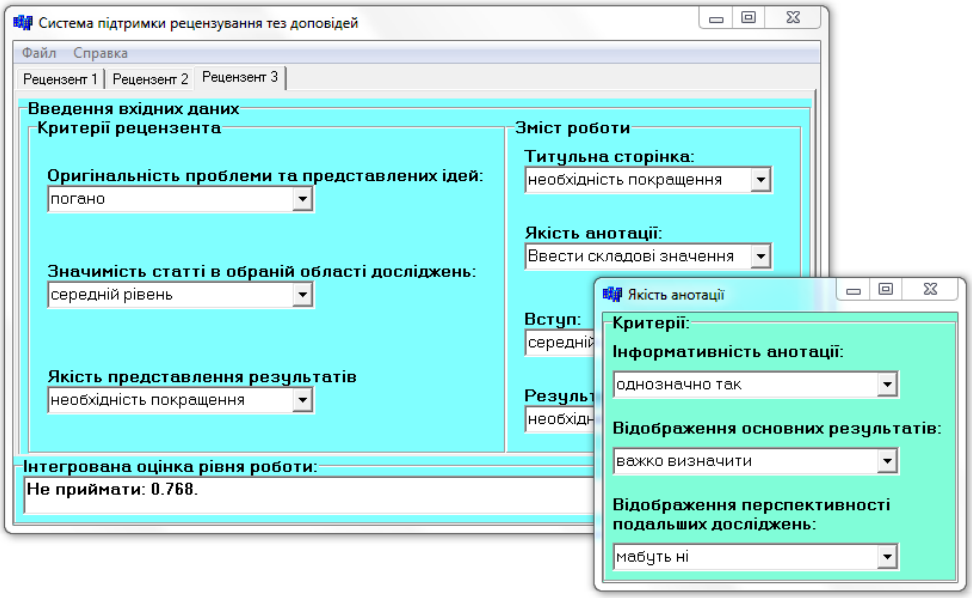

Рис. 3. Графічний інтерфейс користувача системи для оцінювання рівня наукових робіт

Якщо використовувати дерево логічного виведення як граф інформаційних залежностей, то доцільно застосувати паралельно-ярусну або динамічну схему обчислень, яка дозволяє скоротити час обчислень. Таке скорочення відбувається за рахунок одночасного виконання логічного виведення певної кількості нечітких систем.

На рис. 4 показано схему процедури run_of_storey_sheme_mpi(), що забезпечує паралельне виконання багаторівневих нечітких систем логічного виведення за допомогою побудови паралельно-ярусної або динамічної схеми та технологічних можливостей MPI.

```
void run_of_storey_sheme_mpi()
{
    storey_sheme( points ); //Побудова дерева ярусів
    ranks( rank_points ); //Визначення номерів процесів
                          // для систем у кожному ярусі
    for( int i = 0; i < points.size(); i++ )
      for( int k = 0; k < points[ i ].size(); k++ )
        if( rank == rank_points[ points[ i ][ k ] ] )
        {  system_number = points[ i ][ k ];
           for( int s = 0; s < N; s++ )
              if( system_r( system_number, s ) )
               {
                  MPI_Recv( &result_value, 1, MPI_DOUBLE,
                           rank_points[ s ], MPI_ANY_TAG,
                           MPI_COMM_WORLD, &status );
                  add_inputs( result_value );
               }
           result = FuzzySystems[ system_number ]->run();
           for( int s = 0; s < N; s++ )
              if( system_s( s, system_number ) )
                 MPI_Send( &result, 1, MPI_DOUBLE,
                  rank_points[ s ], 0, MPI_COMM_WORLD );
        }
}
```

Рис. 4. Фрагмент ярусно-паралельної схеми обчислень на основі MPI

Для ярусно-паралельної моделі обчислень важливим є той факт, що операції, яким відповідають вершини одного ярусу, не залежать одне від одного (не перебувають у відношенні зв'язку) [8]. Тому існує паралельна реалізація алгоритму, в якій вони можуть бути виконані паралельно на різних процесорах суперкомп'ютера СКІТ-4 [9]. Тому ярусно-паралельна форма алгоритму може бути використана для підготовки такої паралельної реалізації алгоритму. Побудова ярусів здійснюється по графу залежностей нечітких систем, до початку паралельного виконання.

Розглянемо роботу даної схеми. При роботі даного методу спочатку необхідно виконати розподіл нечітких систем по ярусам за допомогою функції storey_sheme(), де її параметром буде покажчик двомірного динамічного списку для зберігання номерів систем по кожному ярусу. Наступним кроком стане виклик функції ranks(), де визначається для кожної системи в кожному ярусі номер процесу, який виконуватиме дану систему. Номер процесу залежить як від кількості систем в ярусі, так і від кількості виділених процесів. Параметром функції ranks() є покажчик одномірного динамічного списку. Робота програми після побудови даних схем виконується циклічно.

Необхідно звернути увагу, що в даній схемі обмін даними між процесорами здійснюється лише у випадку, якщо системи наступного ярусу, які залежать від системи (блоку правил) поточного ярусу, не будуть обчислені на тому процесорі, на якому була обчислена поточна нечітка система. Інакше отримані значення вихідних змінних зберігаються в оперативній пам'яті та використовуються у наступних ярусах при обчисленні блоків правил, що залежать від даних параметрів. Такий підхід зменшує кількість обмінів між процесами, внаслідок чого скорочується час виконання програми.

За реалізацію вищезазначених рішень відповідають функції system_r() та system_s(), що показані на рис. 4. Функція system_r() перевіряє дані, що мають бути прийняті до початку виконання системи, system_s() – перевіряє необхідність передачі даних системам вищого ярусу, що залежать від поточної нечіткої системи. Запускається подвійний цикл, що обходить кожний ярус, та кожний елемент в ярусі. При цьому для кожного елементу(points[i][k]), номер процесу (rank_points) якого збігається з наявним номером процесу, виконується наступне:

1) зберігається значення елементу points[i][k], що містить номер системи, в окрему змінну system_number;

2) циклічно перевіряється залежність даної системи від інших за допомогою функцій system_r(), та при кожному знаходженні такої залежності за допомогою MPI_Recv() приймаються необхідні дані. Ці дані зберігаються викликом функції add_inputs();

3) викликається виконання нечіткої системи по заданому номеру функцією run(), результат зберігається до змінної result;

4) за допомогою функції system_s() здійснюється перевірка необхідності передачі даних;

5) за допомогою функції MPI_Send()результат виконання даної системи надсилається процесам, які виконують наступні системи, що залежні від даної, якщо такі існують.

На рис. 5 показано динамічну схему паралельних обчислень, що має назву "майстер-робітники". Процедура "майстер-робітники" є схемою паралельного програмування на основі технології MPI для досягнення максимальної продуктивності роботи всіх задіяних процесів за рахунок зменшення їх простою. Основна концепція даної моделі полягає в наступному: одному в процесів (першому) надається статус "майстра" та його завданням є керувати роботою інших процесів. Керування процесами полягає в тому, щоб вибирати завдання, доступне для його виконання в даний момент, та вибирати робочі процеси в якості "робітників", яким надавати виконання таких завдань. Також керуючий процес займається роботою з даними. Вхідні дані відправляє для обробки робочим процесам та отримує від них вихідні дані. Завдання ж кожного робочого процесу – виконувати обчислення нечіткої системи в даний момент часу, наданої керуючим процесом на основі ним же наданих вхідних даних. По закінченню цих обчислень, відправити отримані вихідні дані керуючому процесу.

У даній схемі massiv_sends – масив даних, що передаються процесами; massiv_recvs – масив даних, які процеси приймають. Слід зазначити, що дані як передаватись так і прийматись можуть лише від керуючого процесу робочим, або навпаки від робочих керуючому. Між робочими процесами в даній моделі дані не передаються та не приймаються.

Схему обчислень можна розділити на дві частини: реалізація керуючого процесу (надано нульовому процесу) та реалізація процесів-робітників (роль всіх інших процесів, крім нульового).

Реалізація керуючого процесу містить наступні кроки:

1) створюються змінні цілого типу system_number та process_number, перша з яких означає номер вибраної системи для виконання, друга – процес, що виконуватиме дану систему;

2) запускається цикл з перевіркою умови існування хоча в одної нечіткої системи на черзі виконання. Перевіряє таку умову функція systems_non_empty();

```
  void run_dynamic_parallel_system( int &size, int &rank )
    {
         double massiv_sends[10];
         double massiv_recvs[10];
        if( rank == 0 ){
             int system_number = 0, process_number = 0;
             enum recv_indexes_of_master { index_of_system, output_value };
             while( systems_non_empty() ) {
                 while( processes_non_empty() )
                 {
                      if( !search( system_number ) )
                          break;
                      get_process( process_number );
                      add_output_data( massiv_sends, system_number );
                      MPI_Send( &massiv_sends, 10, MPI_DOUBLE,
                                 process_number, 0, MPI_COMM_WORLD );
                 }
                  if( !MPI_Recv( &massiv_recvs, 10, MPI_DOUBLE,
                                  MPI_ANY_SOURCE, MPI_ANY_TAG, MPI_COMM_WORLD,
                                  &status ) )
                      output_data( massiv_recvs, status );
             }
             exit_system();
        }
        else {
            while( true ) {
                MPI_Recv( &massiv_recvs, 10, MPI_DOUBLE,
                            0, MPI_ANY_TAG, MPI_COMM_WORLD, &status );
                 if( int( massiv_recvs[ 0 ] ) )
                      return;
                 input_data( massiv_recvs );
                 double out = FuzzySystems[ massiv_recvs[ 1 ] ] -> run();
                  massiv_sends[ 0 ] = massiv_recvs[ 1 ];
                  massiv_sends[ 1 ] = out;
                  MPI_Send( &massiv_sends, 10, MPI_DOUBLE, 0, 0, MPI_COMM_WORLD );
             }
         }
    }
};
```

Рис. 5. Фрагмент динамічної схеми "майстер-робітники" паралельних обчислень на основі MPI

3) за наявності систем на виконання, запускається другий цикл, що перевіряє, чи є вільні процеси для виконання обчислень;

4) якщо вільні процеси є - перевіряється наявність незалежної системи для виконання. У разі її існування індекс її записується в змінну system_number. У іншому випадку робота циклу переривається;

5) здійснюється виклик функції get_process(), що записує до її аргументу process_number номер вільного процесу;

6) наступний виклик функції add_output_data() забезпечує підготовку та збереження в масиві massiv_sends даних для передачі їх робочому процесу. До цих даних відносяться: умова закінчення роботи, індекс системи, що необхідно обрахувати, кількість вхідних значень та вхідні дані для системи під вказаним індексом;

7) викликом MPI_Send() здійснюється передача даних масиву massiv_sends від керуючого процесу робочому процесу, що є за номером process_number;

8) після невиконання умови 3), виконується отримання вихідних значень нечітких систем від робочих процесів викликом функції MPI_Recv(). В масиві massiv_recvs, де записується отримана інформація міститься індекс виконаної системи та її вихідне значення. Функція MPI_Recv() є блокуюча, тобто робота керуючого процеса припиняється до отримання повідомлення від будь-якого робітника;

9) отримані дані обробляються функцією output_data(). Дана функція видаляє вже розраховані системи зі списку та позначає процес, від якого прийшло повідомлення як вільний;

10) виконується наступна ітерація циклу 2);

11) якщо умова 2) не виконується, викликається функція exit_system(). Ця функція передає команду всім робочим процесам на завершення, після чого робота даної моделі припиняється загалом.

Розглянемо реалізацію схеми обчислень, що відповідає за робочі процеси. Така схема виконується наступним чином для будь-якого процесу, відмінного від нульового:

1) запускається нескінчений цикл;

2) для кожного процесу, якому було передано повідомлення від керуючого процесу викликається функція прийому даних MPI_Recv(). До масиву massiv_recvs записуються такі значення (по порядку): ідентифікатор виходу з підпрограми; індекс системи на виконання; кількість вхідних значень заданої нечіткої системи; вхідні значення;

3) Перевіряється ідентифікатор виходу з підпрограми та якщо він встановлений як істинний, то вихід з підпрограми виконується;

4) викликом функції input_data() відбувається зчитування вхідних даних для їх подальшої обробки роботою нечіткої системи;

5) викликається метод run(), що виконує обчислення нечіткої системи із заданим на вході індексом цієї системи. Згідно індексу задана система вибирається зі списку систем FuzzySystems. Отриманий вихідний результат записується у змінну out;

6) формується масив даних massiv_sends для відправлення їх керуючому процесу. Відправляється індекс нечіткої системи, що була обчислена та її вихідне значення.

Отримані результати прискорення у випадку показаної на рис. 2 багаторівневої нечіткої системи оцінювання рівня наукових робіт показані на рис. 6. Очевидно, що при використанні до шести процесорів більшу ефективність демонструє паралельно-ярусна система. Це відбувається за рахунок того, що в паралельно-ярусній системі задіяні всі процеси для розрахунків даних. В динамічній же системі, як вже згадувалось вище, один процес завжди задіяний лише в управлінні іншими процесами, і не приймає в обробці даних ніякої участі. При збільшені кількості процесів від 8 до 16, більшого прискорення набуває саме динамічна система за рахунок відсутності в ній ярусів. Ярусно-паралельна схема є менш ефективною при великій ширині окремих ярусів, зокрема, першого, ширина якого становить 18 систем. Алгоритм ярусно-паралельної схеми розбиває нижній ярус на декілька підярусів, обчислюючи їх послідовно, що звичайно веде до збільшення часу виконання. Збільшення кількості процесорів до 18 призводить до невеликої але переваги ярусно-паралельної моделі. Враховуючи те, що час виконання однієї нечіткої системи приблизно однаковий, яруси рівномірно виконуються один за одним. Зменшення часу виконання ярусно-паралельної системи в порівнянні з динамічною досягається зменшенням кількості обміну даними між процесами. Адже динамічна схема передбачає, що для кожної системи, що виконується, повинно бути здійснено два обміни даними між керуючим процесом і процесом, що обчислює дану нечітку систему. При кількості процесів більшій за 18, ярусно-паралельна схема не дає прискорення, а динамічна найбільше прискорення має при використанні 20 процесів. Це відбувається тому, що 18 процесах, кількість тих з них, що задіяні в обчисленнях в динамічній моделі дорівнює 17, тобто ця схема ще не може обрахувати паралельно одразу всі 18 наявних нечітких систем, але може це здійснити при 20 процесах.

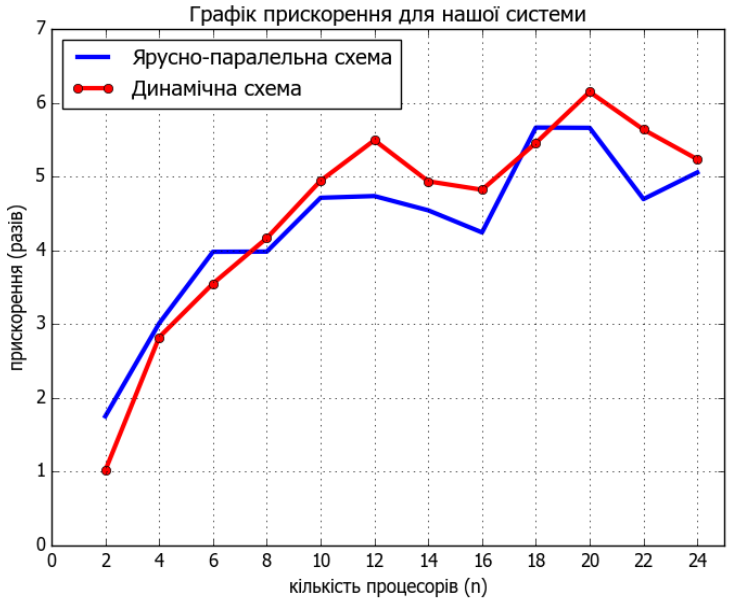

Рис. 6. Прискорення ярусно-паралельної та динамічної схем обчислень для багаторівневої нечіткої системи

Таким чином, при обчисленні представленої нечіткої багаторівневої системи найвищі показники прискорення становлять: для ярусно-паралельної системи $\approx 5{,}7$, для динамічної $\approx 6{,}1$.

В загальному випадку, модель багаторівневої нечіткої системи може бути представлена ациклічним графом, вершини якої виконують окремі блоки нечітких правил. Ефективність ярусно-паралельних обчислень для такої системи оцінювалася за допомогою генерації орієнтованого ациклічного графа залежностей між нечіткими системами, який не містить циклів, та наявність ребер якого визначалася за допомогою датчика випадкових чисел. На рис. 5 наведений отриманий час виконання багаторівневих нечітких систем на кластері СКІТ для виконання багаторівневої нечіткої системи для кількості вершин (v=1000), де v–кількість вершин.

Важливим параметром при цьому є кількість ребер, яка визначалася як $e = v^q$. Для графу з кількістю вершин v=1000, $q = 1$ при виконанні на 24 вершинах СКІТ-4 отримано прискорення: для ярусно-паралельної моделі $\approx 13{,}2$, для динамічної моделі $\approx 18{,}6$. Аналогічне прискорення при кількості $1000^{1{,}7}$ ребер складає 2,8 для обох типів моделей розпаралелювання. Таким чином, при збільшенні кількості ребер, збільшується час виконання як ярусно-паралельної моделі, так і моделі майстер-робітники, оскільки, незалежно від кількості процесорів, кількість систем, що можуть бути виконані паралельно, зменшується за рахунок великої зв'язності даних систем між собою. Також у цьому випадку час виконання збільшуються за рахунок більшого обсягу передач даних між процесорами.

Отже, досить суттєву різницю в прискоренні можна пояснити тим, що у динамічної системи відсутнє поняття ярусу. Одразу після закінчення обчислень однієї нечіткої системи, кожен робочий процес може приступити до виконання наступної, не чекаючи остаточного виконання усіх систем в ярусі. З цього можна зробити висновок, що при достатній кількості задіяних процесорів та великій кількості з'язків між системами, можна використовувати обидві схеми розпаралелювання: ярусно-паралельну та динамічну. Якщо кількість ребер графу нечіткої системи невелика, то доцільно використовувати саме динамічну схему обчислень, що дозволяє отримати значене прискорення обчислень. Очевидно, що при невеликій кількості процесорів, більш вагоме прискорення отримується з використанням ярусно-паралельної схеми (рис. 6 та 7).

На рис. 7 показано результати прискорення для нечіткої системи Такагі – Сугено, залежності між підсистемами якої згенеровано випадковим чином (ребра графу залежностей). Графік прискорення при $q = 1.7$ (рис. 7) відображає тенденцію, що при кількості процесорів більшій ніж 6, суттєвого покращення ефективності не спостерігається. Така тенденція виникає внаслідок зменшення ширини кожного ярусу, яка стає значно меншою за кількість виділених ресурсів (процесорів). Тому при збільшенні кількості процесорів все більша їх кількість не приймає участі в обчисленнях. При $q = 1$ спостерігається ще більше прискорення ніж при $q = 1.7$. Тому в такому випадку є доцільним подальше збільшення кількості процесорів.

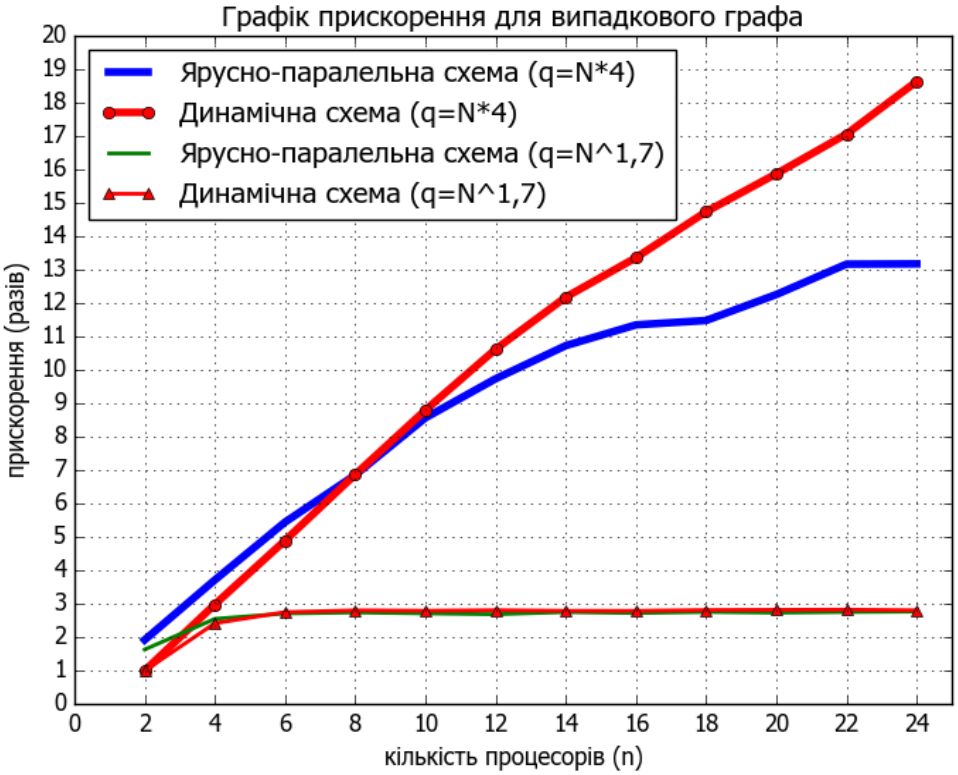

Рис. 7. Прискорення ярусно-паралельної та динамічної схем обчислень для нечіткої системи з великою кількістю підсистем (v=1000)

Висновки

Таким чином, нами запропоновано та обґрунтовано ярусно-паралельну та динамічну моделі для здійснення нечіткого логічного виведення в експертно-діагностичних програмних системах, бази знань яких ґрунтуються на блоках взаємопов'язаних нечітких правилах. Розроблена ярусно-паралельна та динамічна процедура нечіткого виведення, які дозволяють прискорити виконання обчислень в програмній системі, що призначена для оцінки якості наукових робіт. Проведені експерименти на кластерному комп'ютері СКІТ, що дозволяють оцінити ефективність обох схем обчислень за наявності складних графів залежностей між блоками нечітких правил Такагі – Сугено. Проведена порівняльна характеристика показників ефективності ярусно-паралельної та динамічної систем за різних умов.

1. *Zadeh L.A*. The concept of a linguistic variable and its application to approximate reasoning // Information Sciences. – 1975. – Vol. 8, № 8. – P. 199–249; P. 301–357.
2. *Рутковская Д*., *Пилиньский М*., *Рутковский Л*. Нейронные сети, генетические алгоритмы и нечеткие системы. – М.: Горячая линия – Телеком, 2004. – 452 с.
3. *Леоненков А.В.* Нечеткое моделирование в среде MATLAB и fuzzyTECH. – СПб.: БХВ, 2005. – 736 с.
4. *Парасюк И.Н., Ершов С.В.* Трансформационный подход к разработке интеллектуальных агентов на основе нечетких моделей // Проблеми програмування. – 2011. – № 2. – С. 62–78.
5. *Ершов С.В.* Модель интеллектуальных агентов, основанная на нечеткой логике высшего типа // Компьютерная математика. – 2012. – № 1. – С. 10–16.
6. *Ершов С.В.* Принципы построения нечетких мультиагентных систем в распределенной среде // Компьютерная математика. – 2009. – № 2. – С. 54–61.
7. *ТААС 2015*. – http://taac.org.ua.
8. *Воеводин В.В., Воеводин Вл.В.* Параллельные вычисления. – СПб.: БХВ-Петербург, 2002. – 608 с.
9. *Суперкомпьютеры* ИК НАН Украины. – http:// /icybcluster.org.ua.

References

1. Zadeh L. A. (1975) The concept of a linguistic variable and its application to approximate reasoning // Information Sciences. – Vol. 8, № 8. – P. 199–249, P. 301–357.
2. Rutkovskaya D., Pilinsky M., Rutkovsky L. (2006) Neural networks, genetic algorithms and fuzzy systems. – Moscow: Goryachaya liniya – Telekom. – 452 p.
3. Leonenkov A.V. (2005) Fuzzy modeling in MATLAB environment and fuzzyTECH. – SPb .: BHV-Petersburg. – 736 p.
4. Parasyuk I.M., Yershov S.V. (2011) Transformational approach to the development of intelligent agents based on fuzzy models // Problems of programming, № 2. – C. 62–78.
5. Yershov S.V. (2012) Model of intelligent agents based on highest type fuzzy logic // Computer mathematics, №1, K .: Institute of Cybernetics Glushkov National Academy of Sciences of Ukraine. – P. 10–16.
6. Yershov S.V. (2009) Principles of construction of fuzzy multi-agent systems in a distributed environment // Computer mathematics, №2, K .: Institute of Cybernetics Glushkov National Academy of Sciences of Ukraine. – P. 54–61.
7. TAAC 2015. – http://taac.org.ua.
8. Voevodin V.V., Voevodin Vl.V. (2002) Parallel computing. - SPb .: BHV-Petersburg. – 608 p.
9. Supercomputers of IC NAS of Ukraine. – http://icybcluster.org.ua.

Про авторів:

Єршов Сергій Володимирович,
доктор фізико-математичних наук, завідувач відділу.
Кількість наукових публікацій в українських виданнях – 63.
Кількість наукових публікацій в іноземних виданнях – 9.
http://orcid.org/0000-0002-9895-777X,

Пономаренко Роман Миколайович,
аспірант.
Кількість публікацій у вітчизняних і закордонних виданнях – 0.
http://orcid.org/0000-0001-9681-2297.

Місце роботи авторів:

Інститут кібернетики імені В.М. Глушкова НАН України.
03680, Київ-187, проспект Академіка Глушкова, 40.
Тел.: (044) 526 6422, (044) 526 5168.
E-mail: sershv@ukr.net,
ponomarenko_roman@ukr.net