УДК 004.94: 004.428

І.В. СТЕЦЕНКО

ПАРАЛЕЛЬНИЙ АЛГОРИТМ ІМІТАЦІЇ ПЕТРІ-ОБ'ЄКТНОЇ МОДЕЛІ

Анотація. Розглянуто ефективний алгоритм імітації дискретно-подійних систем з великою кількістю елементів. Із застосуванням Петрі-об'єктного моделювання та паралельних обчислень розроблено алгоритм, який одночасно відтворює функціювання Петрі-об'єктів моделі в окремих потоках. Лінійну залежність часу його виконання від складності моделі підтверджено результатами експериментальних досліджень.

Ключові слова: стохастична мережа Петрі, алгоритм імітації, паралельні обчислення.

ВСТУП

Інформаційні системи поступово розширюють свою дію від простих процесів накопичення та зберігання даних до процесів управління організацією чи бізнес-системою, від управління потоками даних (workflow management) до управління бізнес-процесами (business process management). Відповідно до цього зростає важливість використання методів аналізу процесів функціювання систем замість методів аналізу даних, а для інформаційних систем, спрямованих на аналіз процесів, введено новий термін «процесно-орієнтовані (process-aware) інформаційні системи» [1]. Основним формалізмом, який застосовують для аналізу процесів, є мережі Петрі [2]. Їхні переваги: зручність графічного представлення, математичний формалізм, високий рівень деталізації процесів функціювання дискретно-подійних систем та різноманітні засоби аналізу.

Алгоритми імітації дискретно-подійних систем використовують як складові компоненти інформаційних систем для оптимізації управлінських процесів, а також ухвалення рішень. Суттєві затримки обчислень в таких системах призводять до негативних наслідків у роботі технічних систем, які вони обслуговують. Моделі сучасних транспортних, виробничих, фінансово-економічних, еколого-економічних систем містять опис сотень елементів і тисяч подій, тому підвищення швидкодії алгоритмів імітації є важливою задачею.

Петрі-об'єктний підхід зменшує час виконання алгоритму імітації за рахунок переходу від перегляду стану елементарних переходів мережі Петрі до перегляду стану груп таких переходів, об'єднаних у змістові об'єкти моделі. Алгоритм імітації Петрі-об'єктної моделі, наведений у [3], є універсальним для класу дискретно-подійних систем. У роботі [4] отримано оцінку складності цього алгоритму аналітичним способом та експериментально доведено її коректність. У цьому дослідженні сформульовано задачу розпаралелювання такого алгоритму.

Безперечними перевагами паралельних обчислень є швидкість, ефективність використання обчислювальних ресурсів, (гіпотетична) можливість досягнення лінійної залежності часу виконання обчислень від складності моделі, можливість реалізації обчислень в розподілених системах. Технології паралельних обчислень потребують спеціального розроблення алгоритму, який поділяє усі операції на ті, що можна виконати послідовно, та ті, що можна виконати паралельно, а також визначає окремо послідовність дій кожного потоку обчислень. У прикладних задачах для досягнення значного ефекту паралельні алгоритми будують з урахуванням специфіки розв'язуваної задачі, використовуючи декомпозицію простору даних задачі (domain decomposition) або функціональну декомпозицію на підзадачі (functional decomposition) [5].

Розроблення паралельних алгоритмів імітації є складною задачею, оскільки потребує контролю за станом моделі в часі. Імітаційна модель виконує упорядковану в часі послідовність подій, і будь-яке порушення цієї послідовності є порушенням логіки функціювання моделі. Відомі підходи до побудови паралельних алгоритмів імітації ґрунтуються на розбитті функціювання моделі на логічні процеси, які взаємодіють [6]. Корегування перебігу процесів здійснюють так, щоб не порушити упорядкованості усіх подій моделі в часі. Оскільки Петрі-об'єктну модель конструюють з об'єктів, динаміка яких описується стохастичною мережею Петрі, її динаміка від початку створення розбита на елементарні процеси і не потребує штучного розбиття для розпаралелювання. Іншою відмінністю побудованого алгоритму паралельної імітації є те, що кожен Петрі-об'єкт використовує локальну змінну часу (замість загальної для всієї моделі), узгоджуючи її просування тільки з об'єктами, які безпосередньо з ним взаємодіють.

ПАРАЛЕЛЬНІ ОБЧИСЛЕННЯ ТА АЛГОРИТМИ ІМІТАЦІЇ

Багатопотокова технологія java має широкі можливості для паралельного програмування: синхронізовані методи та фрагменти програм, засоби взаємодії потоків, блокувальні об'єкти [7]. Бібліотеки паралельного програмування містять певні засоби автоматизації паралельних обчислень, такі як розпаралелювання дій циклу або паралельне сортування елементів масиву. Проте їхнє безпосереднє застосування до алгоритму імітації не призводить до значного ефекту і навіть сповільнює його роботу. Це пояснюється тим, що в алгоритмі імітації такі дії часто повторюються (при кожному просуванні модельного часу), і кожного разу знову створюються нові потоки для кожного розпаралеленого циклу чи розпаралеленого сортування. У результаті час, який витрачається на створення потоків, не компенсується пришвидшенням, отриманим за рахунок розпаралелювання.

У роботі [8] наведено класифікацію наявних способів розпаралелювання алгоритмів імітації в залежності від рівня, на якому застосовують паралелізм: програмний застосунок (application level), функціювання імітаційної моделі (functional level), її алгоритмізація (event level). Усі три способи застосовують на різних етапах процесу моделювання: на етапі виведення результатів моделювання при програмуванні візуальних середовищ імітаційного моделювання, на етапі експериментування з імітаційною моделлю при проведенні повторюваних прогонів моделі, на етапі реалізації моделі при програмуванні імітаційного алгоритму. Перші два способи ефективно використовують у прикладних програмах, а третій вимагає спеціального розроблення програми, що враховує особливості конкретної моделі. При паралелізмі імітаційного алгоритму модель розбивають на фрагменти, які можуть певний час виконуватись незалежно один від одного, і запускають їх на одночасне виконання.

Контроль за просуванням часу в імітаційних алгоритмах потребує розроблення механізмів синхронізації. У роботі [9] наведено консервативний та оптимістичний [6] способи. В обох підходах фрагменти моделі обмінюються між собою повідомленнями про час наступної найближчої події. При консервативному способі кожен фрагмент перебуває в очікуванні, доки від інших не надійшла інформація про час наступної події, тобто синхронізація відбувається за рахунок очікування, що призводить до неефективного використання паралелізму. При оптимістичному способі фрагмент повертається у відповідний минулий стан, якщо від іншого фрагменту надійшла інформація про подію, яку виконано до поточного моменту часу, тобто синхронізація відбувається за рахунок скасування виконаної події, що потребує запам'ятовування попереднього стану елементів моделі. Такий спосіб має перевагу стосовно часу виконання, але є менш ефективним з точки зору використання ресурсів пам'яті.

АЛГОРИТМ ІМІТАЦІЇ ПЕТРІ-ОБ'ЄКТНОЇ МОДЕЛІ

Петрі-об'єктна модель складається з об'єктів, динаміку яких описують стохастичною мережею Петрі з багатоканальними та конфліктними переходами. Математичні рівняння такої мережі виведено в [10] і доведено, що вони містять як окремий випадок рівняння детермінованої мережі Петрі та рівняння базової мережі Петрі. Поєднання Петрі-об'єктів у моделі відбувається двома можливими способами: з використанням спільної позиції або ініціалізації подій. Як доведено в [11], таке з'єднання забезпечує опис динаміки усієї моделі стохастичною мережею Петрі. Цей факт гарантує обчислюваність моделі, а з математичних рівнянь стохастичної мережі Петрі випливає оцінка обчислювальної складності Петрі-об'єктної моделі у такому вигляді [4]:

$$O(n \cdot q \cdot v \cdot time \cdot (v \cdot (n \cdot v + k^2 + k + v) + q^2 + q)), \qquad (1)$$

де *time* — час моделювання, n — середня кількість переходів одного Петрі-об'єкта, v — середня кількість активних каналів переходу мережі Петрі за одиницю часу, q — кількість Петрі-об'єктів, k — середня кількість конфліктних переходів одного Петрі-об'єкта.

У роботі [4] визначено оцінку обчислювальної складності Петрі-об'єктної моделі та наведено результати експериментальних досліджень, що її підтверджують. Для великих систем кількість переходів одного Петрі-об'єкта значно менша за кількість об'єктів моделі, тобто $n << q$. Окрім того, завжди виконано $k \le n$. Тому для великих систем вираз (1) зростає не швидше ніж $O(n \cdot q^3 \cdot v^2 \cdot time)$. Отже, обчислювальна складність Петрі-об'єктної моделі зростає пропорційно до кубу кількості її Петрі-об'єктів.

Послідовний алгоритм імітації Петрі-об'єктної моделі без конфлікту об'єктів складається з таких дій:

1) визначити Петрі-об'єкт $O_{\min}$ з найменшим значенням часу найближчої події $t_{\min}$;

2) просунути модельний час у момент найближчої події $t_{\min}$ і виконати подію: вихід маркерів з переходів Петрі-об'єкта $O_{\min}$, вхід маркерів в усі Петрі-об'єкти моделі;

3) дії 1, 2 повторювати, доки модельний час менший за кінець інтервалу моделювання.

Якщо вихід з різних Петрі-об'єктів здійснюється одночасно у спільну позицію, яка є вхідною для інших Петрі-об'єктів, то виникає конфлікт об'єктів, оскільки від того, який саме Петрі-об'єкт виконає першим вихід маркерів, залежить подальше функціювання моделі. Конфлікт розв'язують вибором одного об'єкта з рівною ймовірністю серед об'єктів з найвищим пріоритетом. Отже, алгоритм імітації Петрі-об'єктної моделі з конфліктом об'єктів складається з таких дій:

1) визначити список K Петрі-об'єктів з найменшим значенням часу найближчої події $t_{\min}$;

2) відсортувати список K за значенням пріоритету та вибрати з рівною ймовірністю серед об'єктів з найвищим пріоритетом об'єкт $O_{\min}$, для якого виконуватиметься подія;

3) просунути модельний час у момент найближчої події $t_{\min}$ і виконати подію: вихід маркерів з переходів Петрі-об'єкта $O_{\min}$, вхід маркерів в усі Петрі-об'єкти моделі.

4) дії 1–3 повторювати, доки модельний час менший за кінець інтервалу моделювання.

Зауважимо, що складова q^2 у виразі (1) присутня через сортування Петрі-об'єктів за пріоритетом. Тому алгоритм імітації Петрі-об'єктної моделі без конфлікту для великих систем має зростання складності пропорційно до квадрату кількості об'єктів.

Таким чином, Петрі-об'єктна модель має алгоритм та структуру, зручні для розпаралелювання: її динаміка відтворюється однотипними фрагментами моделі, що діють за загальними для всіх фрагментів правилами. Як випливає із закону Амдала, ефективність розпаралелювання залежить від співвідношення частин паралельного та послідовного виконання у побудованому алгоритмі. Звичайний алгоритм імітації організований таким чином, що при кожному просуванні часу здійснюється одна або декілька подій. Оскільки подій, які виконуються одночасно, як правило, небагато, спрямовувати зусилля на розпаралелювання дій, що реалізують події в кожний конкретний момент часу, неефективно. Потрібно організувати імітацію моделі таким чином, щоб відбувалася одночасна імітація фрагментів моделі.

ПАРАЛЕЛЬНИЙ АЛГОРИТМ ПЕТРІ-ОБ'ЄКТНОГО МОДЕЛЮВАННЯ

У реальному світі об'єкти змінюються в часі відповідно до своєї динаміки одночасно, не зважаючи один на одного, аж доки не виникне подія, в якій вони задіяні спільно. Звідси надамо можливість кожному Петрі-об'єкту функціювати незалежно в межах, доки на його функціювання не впливають події в інших Петрі-об'єктах. Розглянемо взаємодію двох Петрі-об'єктів: A і B (рис. 1). Введемо означення.

Означення 1. Перехід мережі Петрі-об'єкта є вихідним, якщо при його запуску здійснюється вихід у спільну позицію з іншим об'єктом або вихід у позицію іншого об'єкта (рис. 1, *а*, *б*).

Означення 2. Позиція об'єкта є вхідною, якщо вона є спільною з позицією іншого об'єкта або в цю позицію здійснюється вихід з переходу іншого об'єкта (рис. 1, *в*, *г*).

Означення 3. Якщо об'єкт A має вихідний перехід, з якого здійснюється вихід маркерів у позицію об'єкта B, то об'єкт B є next-об'єктом для об'єкта A.

Означення 4. Якщо об'єкт B має позицію, в яку здійснюється вихід маркерів з вихідного переходу об'єкта A, то об'єкт A є previous-об'єктом для об'єкта B.

Означення 5. Безпечним інтервалом імітації об'єкта є інтервал часу між двома послідовними виходами маркерів з вихідного переходу його previous-об'єкта.

Таким чином, для кожного об'єкта мають бути визначені його previous-об'єкти та next-об'єкти. Якщо відповідного зв'язку немає, то в об'єкті вказують посилання на «нульовий» previous- або next-об'єкт (null object). На інтервалі часу

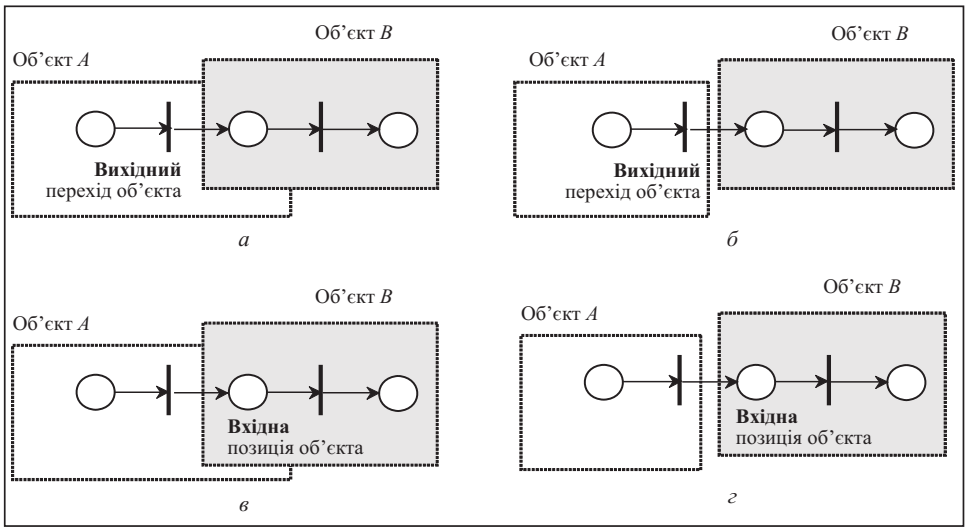

Рис. 1. Схема взаємодії Петрі-об'єктів: вихід маркерів з переходу об'єкта у спільну позицію його next-об'єкта (*а*); вихід маркерів з переходу об'єкта у позицію його next-об'єкта (*б*); вхід маркерів у спільну позицію об'єкта з його previous-об'єкта (*в*); вхід маркерів у позицію об'єкта з переходу його previous-об'єкта (*г*)

між послідовними входами маркерів ззовні динаміка об'єкта не залежить від функціювання його previous-об'єктів, тому події у межах цього інтервалу можуть відбуватись в об'єктах незалежно одна від одної, а значить, паралельно. Припустимо, що модель побудовано таким чином, що для довільної пари об'єктів A і B об'єкт A не може бути одночасно previous-об'єктом і next-об'єктом для об'єкта B. Побудуємо паралельний алгоритм Петрі-об'єктної моделі за такими правилами.

1. Кожний Петрі-об'єкт здійснює імітацію в локальному часі в окремому потоці.

2. Кожний об'єкт, який має next-об'єкт, передає йому інформацію щодо моментів виходу маркерів з об'єкта і призупиняє своє функціювання при значному накопиченні такої інформації.

3. Кожний об'єкт, який має previous-об'єкт, виконує імітацію в межах до наступної події входу в нього маркерів з previous-об'єкта, поступово просуваючи свій локальний час до повного вичерпання накопиченої інформації про вхідні події або очікує надходження такої інформації зі свого previous-об'єкта.

4. Об'єкт, який має previous-об'єкт, при досягненні моменту часу входу маркерів в об'єкт відновлює вихід маркерів у спільну позицію з переходу previous-об'єкта та продовжує імітацію.

5. Об'єкт, який має next-об'єкт, при досягненні моменту виходу маркерів з об'єкта призупиняє вихід маркерів у позицію next-об'єкта. Цей вихід відновить next-об'єкт у відповідний момент часу свого функціювання.

6. Об'єкт, який вичерпав час моделювання, передає повідомлення про це next-об'єкту, якщо такий є, і завершує роботу. Одночасно із завершенням роботи об'єкта завершує роботу і генерований ним потік.

7. Об'єкт, який отримав від previous-об'єкта сигнал про завершення його роботи і водночас вичерпав усі накопичені події, припиняє роботу.

Отже, функціювання кожного об'єкта відбувається у тісній взаємодії з його previous та next-об'єктами, але незалежно від інших об'єктів, що дозволяє кільком об'єктам одночасно здійснювати імітацію, не порушуючи логіки функціонування один одного. Узгодження подій в різних об'єктах відбувається за рахунок завдання безпечних інтервалів часу, в межах яких функціювання об'єкта не залежить від інших об'єктів.

Для кожного об'єкта введемо буфер зовнішніх подій, який зберігає моменти часу надходження маркерів з інших об'єктів, що не опрацьовані на поточний момент часу. У межах від одного моменту зовнішньої події до наступної гарантовано не виникає зовнішніх подій і об'єкт може здійснювати імітацію своїх внутрішніх подій. Порожній стан буфера означає, що наступний момент зовнішньої події невідомий (на поточний момент виконання алгоритму імітації). Стан буфера, в якому останній його елемент є нескінченність (максимально допустиме число), означає, що previous-об'єкт завершив свою роботу і надходження зовнішніх подій не очікується.

Локальний час об'єкта просувається за такими правилами:

- якщо час найближчої події менший за межу безпечного інтервалу, то просунути локальний час у момент найближчої події, виконати (внутрішні) події, для яких час збігається з поточним моментом, та продовжити імітацію об'єкта;
- інакше
  - якщо межа безпечного інтервалу менша за межу інтервалу моделювання, то просунути локальний час на межу безпечного інтервалу та виконати зовнішню подію (відновлення маркерів у вхідних позиціях об'єкта);
  - інакше просунути локальний час на межу безпечного інтервалу та завершити імітацію об'єкта.

Межа безпечного інтервалу визначається за наступними правилами:

- якщо немає previous-об'єкта, то межа встановлюється в час моделювання;
- інакше
  - якщо буфер зовнішніх подій не порожній і перша подія менша за час моделювання, то межа встановлюється в момент першої зовнішньої події;
  - інакше — в час моделювання.

Об'єкт перебуває в очікуванні (призупиняє функціювання), якщо і тільки якщо:

- він перебуває у stop-стані: кількість маркерів недостатня для запуску переходів, момент виходу маркерів з переходу більший за границю безпечного інтервалу і, якщо є previous-об'єкт, перша подія в буфері менша за границю безпечного інтервалу;
- є previous-об'єкт і буфер зовнішніх подій порожній;
- є next-об'єкт і його буфер зовнішніх подій перевищує заданий ліміт.

Об'єкт розблоковується (припиняє очікування), якщо і тільки якщо:

- його previous-об'єкт здійснив вихід у позицію, спільну з даним об'єктом;
- його next-об'єкт подав сигнал про відновлення роботи об'єкта у разі, якщо він вичерпав буфер подій до заданого ліміту.

Об'єкт завершує імітацію, якщо досягнутий кінець інтервалу моделювання.

Синхронізація виходу маркерів у спільну позицію відбувається таким чином:

- у момент здійснення виходу маркерів з вихідного переходу Петрі-об'єкта вихід маркерів у позицію next-об'єкта не здійснюється, замість цього поточний момент часу запам'ятовується у буфері зовнішніх подій next-об'єкта;
- коли локальний час об'єкта досягає першого значення буфера подій, здійснюється вхід маркерів у вхідну позицію об'єкта, а перше значення з буфера зовнішніх подій видаляється.

Якщо в буфері об'єкта першим елементом є нескінченність, це означає, що його previous-об'єкт завершив імітацію на інтервалі імітації і зовнішніх подій більше не очікується. Об'єкт продовжує імітацію до завершення інтервалу моделювання без очікування зовнішніх подій.

Якщо об'єкт має декілька previous-об'єктів, то для кожного з них створюється свій буфер подій. Момент найближчої події можна визначити тільки тоді, коли надійшла інформація про наступні події від усіх previous-об'єктів. Тому робота такого об'єкта призупиняється, якщо хоча б один буфер подій порожній, і відновлюється, якщо всі буфери подій не порожні. Щоб запобігти взаємному блокуванню об'єктів, потрібно відстежувати стан, в якому об'єкт і його previous-об'єкти одночасно перебувають у стані очікування (коли хоча б один з їхніх буферів подій є порожнім). У цьому випадку потрібно визначити найближчу подію для об'єкта і його previous-об'єктів, просунути локальний час в момент найближчої події і продовжити моделювання.

Якщо об'єкт має декілька next-об'єктів, то його робота призупиняється при досягненні ліміту буфера подій хоча б в одному з його next-об'єктів, і відновлюється, якщо в усіх next-об'єктах кількість подій в буфері подій менша за вказаний ліміт.

Алгоритм, описаний вище, базується на таких припущеннях: 1) модель має хоча б один об'єкт, для якого не заданий previous-об'єкт, і хоча б один об'єкт, для якого не заданий next-об'єкт; 2) для будь-якого об'єкта A всі об'єкти моделі можна поділити на дві неперетинні підмножини: об'єкти, які є його previous-об'єктами або previous-об'єктами його previous-об'єктів, та об'єкти, які є його next-об'єктами або next-об'єктами його next-об'єктів. У реальних умовах перше припущення відповідає відкритій системі, а другого можна досягти, конструюючи модель з Петрі-об'єктів, що не мають зворотних зв'язків.

РЕАЛІЗАЦІЯ ПАРАЛЕЛЬНОГО АЛГОРИТМУ ІМІТАЦІЇ

Для реалізації алгоритму використовуємо багатопотокову технологію java та бібліотеку java.util.concurrent, що надає можливість гнучкого управління потоками. Метод main() програми створює Петрі-об'єктну модель і запускає кожний її Петрі-об'єкт в окремому потоці. При запуску Петрі-об'єкта в його методі run() визначається межа безпечного інтервалу та здійснюється запуск імітації на безпечному інтервалі, доки не досягнутий кінець інтервалу моделювання. Імітація Петрі-об'єкта призупиняється, якщо буфер його зовнішніх подій порожній або буфер зовнішніх подій його next-об'єкта перевищив заданий ліміт. При досягненні межі безпечного інтервалу здійснюється відповідна зовнішня подія та її видалення з буфера зовнішніх подій.

Позначимо:

timeLocal — локальний час Петрі-об'єкта,
timeMod — кінець інтервалу моделювання,
timeLimit — границя безпечного інтервалу імітації Петрі-об'єкта,
timeMin — момент найближчої події,
firstTimeExternalInput — перший момент в буфері зовнішніх подій,
isEmpty — буфер зовнішніх подій порожній,
input() — вхід маркерів в переходи мережі Петрі-об'єкта,
output() — вихід маркерів з переходів мережі Петрі-об'єкта,
goUntil(t) — імітація Петрі-об'єкта на інтервалі часу від моменту timeLoc до моменту t, moveTimeLocal() — просування часу об'єкта.

Метод run() Петрі-об'єкта містить такі дії:

- доки timeLocal < timeMod:
  - якщо є previous-об'єкт,
    - очікувати, доки isEmpty;
    - timeLimit = firstTimeExternalInput;
    - якщо timeLimit > timeMod, то timeLimit = timeMod;
  - інакше timeLimit = timeMod;
  - якщо timeLocal < timeLimit, виконати goUntil(timeLimit);
  - інакше завершити метод run().
- кінець циклу доки.

Метод goUntil(t) здійснює імітацію об'єкта в межах безпечного інтервалу, призупиняючи її, якщо буфер зовнішніх подій порожній:

- доки timeLocal < timeLimit:
  - очікувати, доки мережа знаходиться у stop-стані;
  - input();
  - якщо timeMin < timeLimit,
    - timeLocal = timeMin;
    - output() і продовжити цикл методу goUntil;
  - інакше
    - якщо timeLimit >= timeMod,
      - timeLocal = timeMod;
      - передати сигнал next-об'єкту про завершення своєї роботи;
    - інакше
      - якщо є previous-об'єкт, який не завершив свою роботу,
        - очікувати, доки буфер подій порожній;
        - timeLocal = timeLimit;
        - виконати зовнішню подію (вхід маркерів у спільну позицію та видалення з буфера зовнішніх подій);
        - відновити дію previous-об'єкта, якщо кількість зовнішніх подій зменшилась до прийнятної;
      - timeLocal = timeLimit, що означає виконання умови виходу з циклу;
- кінець циклу доки.

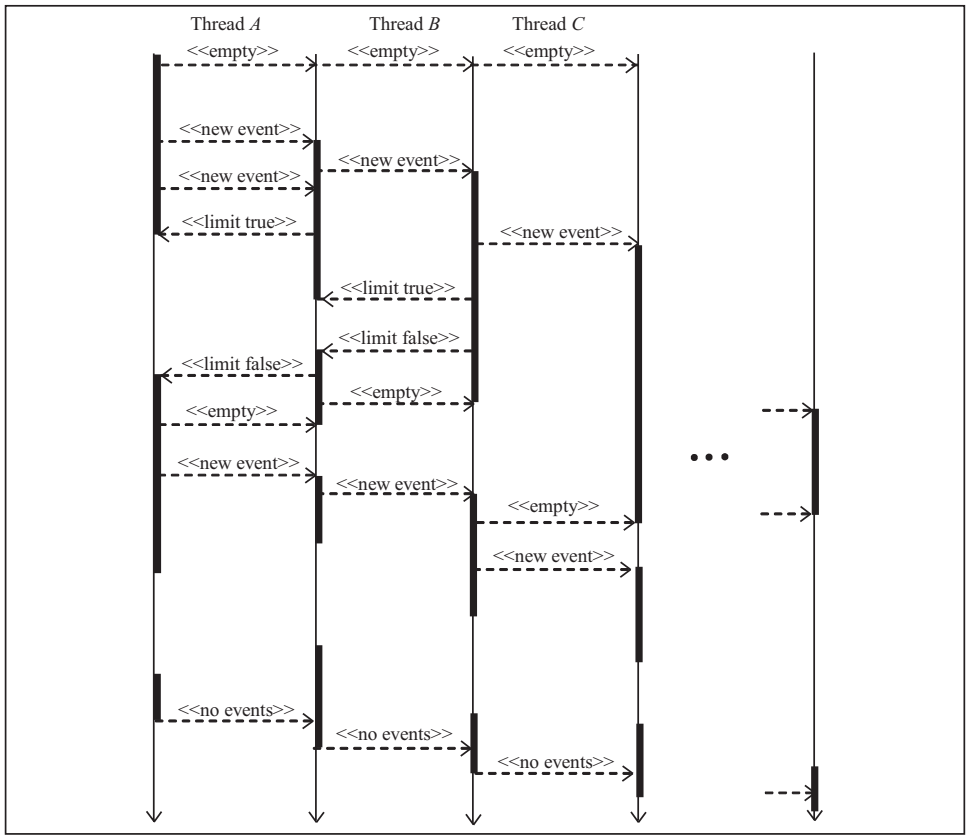

Рис. 2. Схема взаємодії потоків паралельного алгоритму імітації Петрі-об'єктної моделі

Метод output() у разі, якщо здійснюється вихід маркерів у вихідну позицію Петрі-об'єкта, запам'ятовує подію у буфері зовнішніх подій next-об'єкта і призупиняє дію об'єкта, якщо буфер зовнішніх подій його next-об'єкта перевищує заданий ліміт. Повідомлення про поповнення буфера зовнішніх подій відновлює дію об'єкта, який перебував в очікуванні.

Взаємодію потоків наведено на діаграмі (рис. 2). Потоки A, B, C ілюструють дію потоків, що запускають імітацію Петрі-об'єктів A, B, C таких, що A є previous-об'єктом для B, B — next-об'єктом для A і previous-об'єктом для C, C — next-об'єктом для B. Пара повідомлень <<new event>> та <<empty>> інформує об'єкт про надходження нової події від previous-об'єкта і, навпаки, що таких подій немає. Повідомлення <<no event>> сповіщає next-об'єкт про завершення імітації об'єктом, а значить, про припинення надходження подій від previous-об'єкта. Пара повідомлень <<limit true>>, <<limit false>> інформує об'єкт про перевищення ліміту буферу зовнішніх подій його next-об'єкту або, навпаки, про те, що такого перевищення немає.

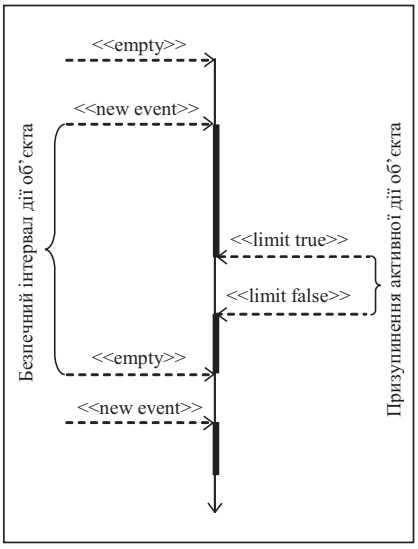

Рис. 3. Схема управління потоком паралельного алгоритму імітації Петрі-об'єктної моделі

Схему управління потоком двома парами повідомлень наведено на рис. 3. Об'єкт

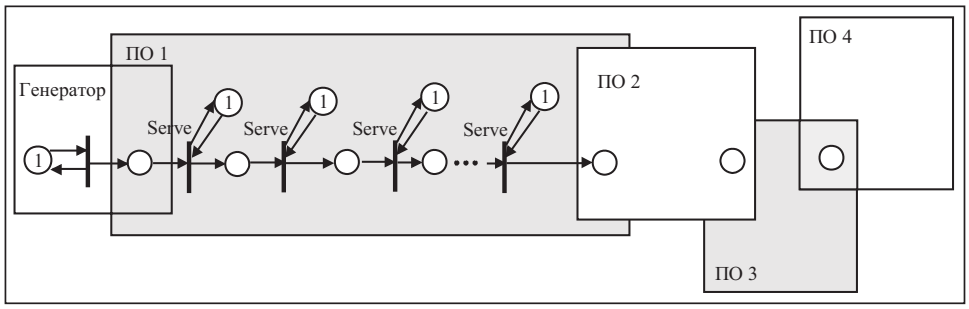

Рис. 4. Схема тестової Петрі-об'єктної моделі мережі масового обслуговування

перебуває в очікуванні, доки його буфер зовнішніх подій порожній, і розпочинає роботу при першому надходженні інформації щодо моменту зовнішньої події. Безпечний інтервал дії об'єкта завершується, якщо його буфер зовнішніх подій порожній. Під час активної дії об'єкта може надійти повідомлення про перевищення ліміту буфера зовнішніх подій його next-об'єкта, яке призупиняє активну дію об'єкта, доки не надійде альтернативне повідомлення.

З опису алгоритму випливає, що на час виконання алгоритму впливатимуть затримки на очікування повідомлень від previous-об'єктів та від next-об'єктів. Останні можна завжди зменшити за рахунок збільшення обмеження на кількість подій в буфері зовнішніх подій, а перші залежать від внутрішньої складності об'єктів і не можуть бути змінені тільки параметрами алгоритму. Отже, моделі з послідовною структурою, в яких затримки на очікування повідомлень від previous-об'єктів більші, матимуть більший час виконання алгоритму. Петрі-об'єкти, які мають однакові previous-об'єкти, працюватимуть в алгоритмі незалежно один від одного та одночасно.

Тестування алгоритму здійснювалось на моделі, яка відтворює імітацію послідовно з'єднаних систем масового обслуговування (СМО). Для такої мережі масового обслуговування можна виконати аналітичні розрахунки і порівняти потім з результатами імітації. Петрі-об'єктна модель складається з об'єктів «Генератор» та «Група СМО» (рис. 4). Петрі-об'єкт «Група СМО» відтворює динаміку послідовно з'єднаних кількох СМО. Кількість СМО, що об'єднані в групу, вказується в конструкторі Петрі-об'єкта.

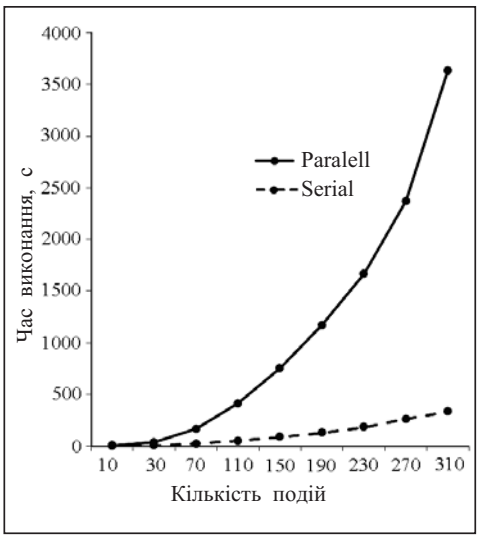

Рис. 5. Графіки ефективності паралельного алгоритму імітації з глобальною змінною часу

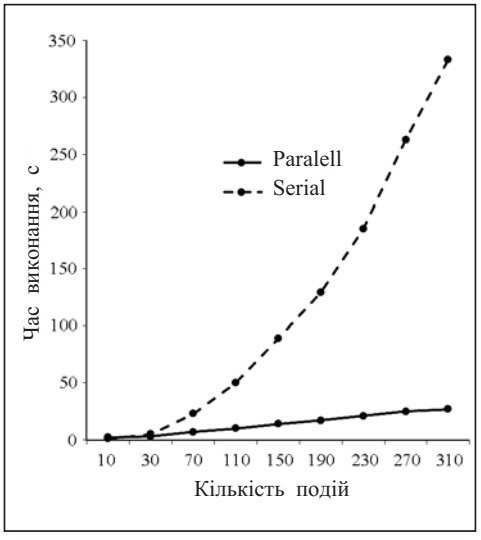

Рис. 6. Графіки ефективності паралельного алгоритму імітації з локальною змінною часу

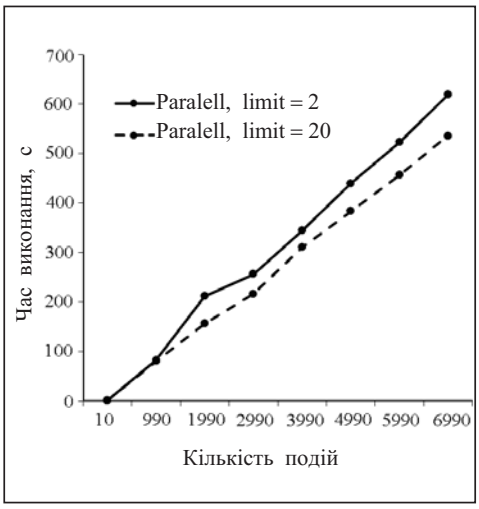

Рис. 7. Графіки зростання часу виконання паралельного алгоритму імітації з локальною змінною часу

За результатами тестових прогонів порівнювали середнє значення черги кожної СМО з розрахованим аналітичним значенням. Спостережувана похибка складає не більше 3%.

Експериментальне дослідження проводилось на двоядерному комп'ютері (Processor Intel® Core™2 Duo CPU E8400, @3.00 GHz, RAM 4 GB). Порівняння часу виконання послідовного та паралельного алгоритму імітації Петрі-об'єктної моделі, побудованого з використанням глобальної змінної часу та найпростіших засобів управління потоками, показало значне збільшення часу виконання паралельного алгоритму при збільшенні кількості подій в моделі (рис. 5). Це пояснюється тим, що глобальна змінна часу та синхронізовані методи запуску подій змушують об'єкти очікувати один одного при кожному просуванні часу. Алгоритм працює правильно, але неефективно.

Навпаки, розроблений паралельний алгоритм імітації Петрі-об'єктної моделі з локальними змінними часу за результатами експериментального порівняння має високу ефективність. При збільшенні кількості подій в моделі, а саме: при кількості подій близько 300, спостерігається зменшення часу виконання у 14 разів і лінійна залежність часу виконання паралельного алгоритму від складності моделі (рис. 6). При подальшому збільшенні кількості Петрі-об'єктів у моделі до 700 і відповідно кількості подій до майже 7000 лінійна залежність часу виконання паралельного алгоритму від складності моделі зберігається. Це означає, що завдяки паралелізму час обчислення Петрі-об'єктної моделі зменшився від кубічної оцінки (1) до лінійної (рис. 7).

ВИСНОВКИ

Ефективний паралельний алгоритм імітації можна розробити тільки в результаті його повної перебудови, що враховує усі особливості організації паралельних обчислень: створення та запуск потоків, блокування їхньої дії та синхронізацію з діями інших потоків. Петрі-об'єктна модель має зручну для організації паралельних обчислень структуру, оскільки складається з елементів, динаміка яких задана стохастичною мережею Петрі. Проте використання стандартних засобів розпаралелювання та глобальної змінної часу призводить до коректного, але нешвидкого алгоритму імітації. Розроблений паралельний алгоритм імітації Петрі-об'єктої моделі одночасно відтворює функціювання об'єктів моделі в окремих потоках з урахуванням динаміки тільки об'єктів, з якими вони безпосередньо зв'язані. Експериментальне дослідження алгоритму доводить його коректність та ефективність. Порівняння з аналітичними моделями масового обслуговування показало достатньо високу точність з величиною похибки, що не перевищує 3%. Дослідження ефективності алгоритму свідчить про близьку до лінійної залежність часу виконання алгоритму від складності моделі.

Застосування паралельних обчислень надало можливість зменшити складність алгоритму імітації від кубічної до лінійної, що надзвичайно важливо при моделюванні великих систем. Оскільки Петрі-об'єктний підхід є універсальним для дискретно-подійних систем, розроблений алгоритм може стати універсальним засобом для паралельної імітації таких систем.

СПИСОК ЛІТЕРАТУРИ

1. Van der Aalst W.M.P. Process-aware information systems: Lessons to be learned from process mining. Trans. on Petri Nets and Other Models of Concurrency II. Special Issue on Concurrency in Process-Aware Information Systems. Berlin; Heidelberg: Springer, 2009. P. 1–26.
2. Van der Aalst W.M.P. Business process management as the “Killer App” for Petri nets. *Software and Systems Modeling*. 2015. URL: http://wwwis.win.tue.nl/~wvdaalst/publications/z1.pdf.
3. Стеценко И.В. Алгоритм имитации Петри-объектной модели. *Математичні машини і системи.* 2012. № 1. С. 154–165.
4. Stetsenko I.V., Dorosh V.I., Dyfuchyn A.Yu. Petri-object simulation: software package and complexity. *Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS)*, *2015 IEEE 8th Intern. Conf. (Warsawa) IEEE*. 2015. Vol. 1. P. 381–385.
5. Foster I. Designing and building parallel programs: Concepts and tools for parallel software engeneering. Boston: Addison-Wesley Longman Publishing Co., 1995. 370 p.
6. Fujimoto R.M. Parallel and distributed simulation. *Proc. of the 2015 Winter Simulation Conference.* (Huntington Beach, Ca.). IEEE, 2015. P. 45–49.
7. Garg V.K. Concurrent and distributed computing in java. Hoboken (N.J.): John Wiley & Sons, Inc., 2004. 305 p.
8. Walter J.C. Parallel simulation of queueing Petri Net models. Karlsruhe Institute of technology. Diploma thesis. 2013. URL: https://sdqweb.ipd.kit.edu/publications/pdfs/walter2013-parallel.pdf.
9. Fujimoto R.M. Parallel and distributed simulation system. *Proc. of the 2001 Winter Simulation Conf.* (Arlington, Va.). P. 147–157.
10. Stetsenko I.V. State equations of stochastic timed Petri nets with informational relations. *Cybernetics and Systems Analysis*. 2012. Vol. 48, N 5. P. 784–797.
11. Стеценко И.В. Теоретические основы Петри-объектного моделирования систем. *Математичні машини і системи.* 2011. № 4. С. 136–148.

Надійшла до редакції 31.08.2016

И.В. Стеценко
ПАРАЛЛЕЛЬНЫЙ АЛГОРИТМ ИМИТАЦИИ ПЕТРИ-ОБЪЕКТНОЙ МОДЕЛИ

Аннотация. Рассмотрен эффективный алгоритм имитации дискретно-событийных систем с большим количеством элементов. С применением Петри-объектного моделирования и параллельных вычислений разработан алгоритм, который одновременно воспроизводит функционирование Петри-объектов модели в отдельных потоках. Линейная зависимость времени его выполнения от сложности модели подтверждена результатами экспериментальных исследований.

Ключевые слова: стохастическая сеть Петри, алгоритм имитации, параллельные вычисления.

I.V. Stetsenko
THE PARALLEL PETRI-OBJECT SIMULATION ALGORITHM

Abstract. The paper considers the development of efficient simulation algorithm for discrete event systems with a great number of elements. With the use of Petri-object simulation and parallel computing, the algorithm is developed that simultaneously reproduces model’s Petri-objects operation in separate streams. The linear dependence of the runtime of the developed algorithm on model’s complexity is confirmed by experimental results.

Keywords: stochastic Petri net, simulation algorithm, parallel computing.

Стеценко Інна Вячеславівна,
доктор техн. наук, професор Національного технічного університету «Київський політехнічний інститут імені Ігоря Сікорського», e-mail: stiv.inna@gmail.com; stiv66@yandex.ua.