УДК 004.7

КОРОЛЬКОВ І.В.

СКОРОЧЕННЯ ДОВЖИН МАРШРУТІВ, ОТРИМАНИХ МЕТОДОМ АЛГОРИТМІЧНОЇ МАРШРУТИЗАЦІЇ, ШЛЯХОМ МОДИФІКАЦІЇ ТОПОЛОГІЧНИХ ДЕРЕВ

Розглянуто метод алгоритмічної маршрутизації як механізм абстракції при моделюванні маршрутизації в обчислювальних мережах. Запропоновано два ітераційних метода поліпшення дерева топології мережі. Перший метод передбачає перевірку кожної дійсної модифікації, після чого модифікація відразу ж приймається або відхиляється. Другий метод включає перевірку вузла та відновлення усіх можливих розірваних зв'язків, після чого модифікація приймається або відхиляється на підставі підсумкової зміни загальної кількості пересилань, необхідних для з'єднання усіх пар окремих вузлів у мережі, далі процедура повторюється для кожного вузла. Проведено аналіз можливості застосування обох методів, а також їх комбінації для модифікації маршрутних дерев мереж різних розмірів.

The Algorithmic Routing method as the abstraction technique for computer networks simulation is considered. Two iterative methods of topology trees improvement are proposed. The first method tests each modification and accepts or rejects it immediately. The second method examines a node, reconnects all possible broken links and after that accepts or rejects the modification based on the overall change in the sum total of the number of hops needed to join all distinct pairs of nodes in the network, and than repeats this for every node. The analysis of routing trees modification for networks of different sizes by both methods and its combination is done.

Вступ

Ключова особливість маршрутизації в Інтернет полягає в тому, що кожен маршрутизатор підтримує свою власну таблицю маршрутизації, навіть якщо вона побудована за допомогою обміну повідомленнями зі своїми сусідами. Тим не менш, на відміну від проектування реального маршрутизатора, при моделюванні можна не брати до уваги деякі наявні обмеження.

Метод алгоритмічної маршрутизації (АМ) було висунуто в [1], далі розвинуто у [2]. Ідея методу заснована на роботі з мережами бінарних дерев [3] та передбачає зниження використання пам'яті; при цьому дещо зростає обчислювальна складність.

Алгоритмічна маршрутизація передбачає накладення вихідної топології мережі на *k*-е дерево. Це дерево може бути сформовано шляхом пошуку в ширину (ПВШ) чи в глибину (ПВГ) мережі. При цьому вибір алгоритму пошуку впливає на точність маршрутизації. Значення κ відповідає

максимальній кількості потомків, які охоплені будь-яким вузлом у дереві. Якщо в мережі наявні петлі, вони руйнуються на етапі відображення.

У [4] нами було запропоновано декілька модифікацій вихідного методу АМ для поліпшення продуктивності та масштабованості моделювання великих обчислювальних мереж. Однак слід відмітити, що отримані маршрути не завжди є найбільш короткими, в них можуть виникати деякі спотворення.

У мережі може існувати безліч петель, при цьому кожна петля робить вклад у збільшення середньої відносної різниці довжин маршрутів *R*. Також важливим є показник довжини петель – взаємний вплив та накладення петель ускладнюють завдання аналізу порівняно з простою кільцевою мережею. На додачу до кількості та розміру петель фактор їх розташування в мережі також має значний вплив на величину *R*. З одного боку, якщо петлі перебувають на периферії дерева, вони будуть впливати тільки на збільшення довжин маршрутів, які закінчуються там. З іншого боку, оскільки більшість маршрутів перетинають магістральні канали мережі, петлі, які перебувають на основних магістралях, будуть справляти значний вплив на *R*. Також, оскільки між будь-якими двома гілками дерева існує тільки один шлях, трафік може концентруватися на деяких лініях зв'язку та маршрутизаторах, у той час як інші будуть простоювати.

Постановка завдання

Вибір кореневого вузла в дереві, отриманому шляхом ПВШ, справляє вплив як на якість отриманих маршрутів, так і на обчислювальну складність операції розрахунку наступного пересилання. Як корінь доцільно обрати вузол, при якому середня глибина дерева буде мінімальною, або перебудувати дерево після його формування.

Нехай H – загальна кількість пересилань, необхідних для з'єднання всіх пар окремих вузлів у мережі:

$$H = \sum_{i=1}^{N} \sum_{j=i+1}^{N} Dist(i,\ j).$$

На прикладі декількох мереж різного розміру в табл. 1 наведені середні значення *H*/*D*. Тут *H*/*D* являє собою відношення загальної кількості пересилань у маршруті до кількості пересилань у найкоротшому маршруті та є показником якості маршрутів у дереві. В усіх випадках пріоритет при формуванні маршрутного дерева методом ПВШ було надано візлам з високою зв'язністю. Відносна якість маршрутів варіюється у межах 1,1 ÷ 1,27 від довжини найкоротшого маршруту. На жаль, явного розв'язку задачі вибору кореневого вузла та порядку ПВШ для мінімізації H не існує.

Табл.1. Приклад значень *H*/*D* для мереж,
які включають від 100 до 6400 вузлів

Кількість вузлів	Середнє H/D
100	1,23
200	1,10
400	1,12
800	1,11
1600	1,16
3200	1,17
6400	1,27

Знаходження оптимального дерева для АМ є окремим випадком більш загальної проблеми теорії графів, відомої як задача оптимально комунікативного зв'язного дерева (Optimal Communication Spanning Tree, OCST), висунута у [5].

Формально: неорієнтований граф G заданий як $G = (V, E)$, де V – множина вершин, а E – множина ребер, які з'єднують усі пари вершин. Кількість вершин позначено як $n = |V|$. Ребро між вершинами v_i і v_j позначено як (v_i, v_j). Зв'язне дерево графа G визначено як $T = (V, F)$, де $F \subseteq E$, $|F| = |V| - 1$ і всі вершини зв'язані.

Розв'язання задачі OCST передбачає знаходження зв'язного дерева для графа G, що відповідає певним вимогам. Матриця запитів $R = (r_{ij})$ визначає кількість трафіка між парами вершин. Матриця R має розмір $n \times n$, а r_{ij} – трафік, який підлягає передачі між вузлами v_i і v_j. Матриця відстаней $W = w_{ij}$ також має розмір $n \times n$ та містить у собі вагові коефіцієнти відстаней між кожною парою вузлів. Нехай $P_{ij}(T) \subseteq F$ - множина ребер, які з'єднують вершини v_i і v_j у дереві T. Вага зв'язного дерева $w(T)$ визначено як

$$w(T) = \sum_{v_i, v_j \in V} \left(r_{ij} \sum_{(v_p, v_q) \in P_{ij}(T)} w_{pq} \right).$$

Дерево T є розв'язком задачі OCST, якщо $w(T) \le w(T')$ для всіх інших зв'язних дерев T'.

Оптимальне маршрутне дерево для АМ є окремим розв'язком задачі OCST. Нехай G_n – граф топології мережі, у якому ребра E_n відповідають лініям зв'язку. У вихідну задачу OCST вводяться додаткові умови:

$|G| = |G_n|$,

$r_{ij} = 1$, якщо $i < j$,

$r_{ij} = 0$, якщо $i \ge j$,

$w_{ij} = 1$, якщо $(v_i, v_j) \in E_n$,

$w_{ij} = \infty$, якщо $(v_i, v_j) \notin E_n$.

Як і раніше, G – неорієнтований граф.

Основна задача OCST є *NP*-повною [6]. Сучасні підходи до розв'язання задачі OCST базуються на еволюційних алгоритмах [7]. Але розв'язки задачі OCST обмежились невеликими графами, які зазвичай містять у собі

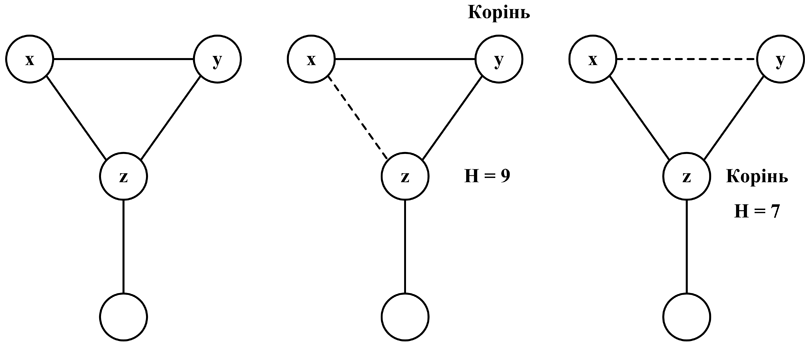
менше ніж 25 вершин [8]. Такі підходи не дають змоги оперувати великими графами, які використовуються в алгоритмічній маршрутизації.

Нещодавні дослідження були спрямовані на пошук алгоритмів для отримання розв'язків, близьких до оптимальних, але запропоновані алгоритми не підходять для графів великих розмірів, оскільки потребують розрахунку найменших маршрутів для всіх пар вузлів, що обчислювально не є доцільним.

Запропонований далі ітераційний метод формування дерев для АМ, на відміну від згаданих вище, не гарантує перебування H у заданих межах, але дає змогу оперувати графами більших розмірів. На кожному кроці проводиться модифікація дерева, метрикою при цьому є H: якщо H зменшується, модифікація зберігається, інакше вона скасовується.

Формування зв'язного дерева для отримання найкоротших маршрутів

Мережа сама по собі не може бути модифікована. Зміні може бути підданий лише спосіб відображення топології на дерево. Якщо мережа містить у собі петлі, то є більше ніж один спосіб відображення на дерево, який впливає на значення H. Розглянемо невелику мережу на рис. 3а. Ця мережа може бути відображена на дерево методом ПВШ двома способами, як показано на рис. 3б і 3в. Обидва дерева дійсні, але мають різні значення H. Проте, призначуючи вузол z замість вузла y предком вузла x, можна перейти від одного дерева до іншого. При цьому зміни не мають уводити петлі чи робити дерево не повністю зв'язаним.



а) б) в)

Рис. 3. Приклад альтернативного вибору кореневого вузла у простій мережі

Як і раніше, $G(V, E)$ – граф топології мережі, а (T, r) – маршрутне дерево, в якому T – зв'язне дерево, а r – кореневий вузол. $S(T, a)$ – множина вузлів у піддереві T з коренем в a. Нехай $x, y, z \in V$, а y – потомок x.

Позначимо модифікацію дерева як

$$T_{new} = \frac{T}{\{(x, y)\} \cup \{(z, y)\}},$$

де $z \in S(T, x)$. Позначимо значення H для дерев T і T_{new} як $H(T)$ і $H(T_{new})$ відповідно.

Розглянемо два ітераційних методи поліпшення дерева АМ. Перший метод передбачає перевірку кожної дійсної модифікації, після чого модифікація відразу приймається або відхиляється. Другий метод включає перевірку вузла та відновлення усіх можливих розірваних зв'язків, після чого модифікація приймається або відхиляється на підставі загальної зміни H, потім процедура повторюється для кожного вузла. Алгоритми цих двох методів наведено на рис. 5 і 6 відповідно.

```
   Задано. Граф G = (V, E). Маршрутне дерево T = (V, F, r), де
F ⊆ E. H(T) – сума відстаней між усіма невпорядкованими парами
вузлів у дереві T.
   Визначити. Маршрутне дерево Tnew, для якого H(Tnew) ≤ H(T).
   (1) Ttest ← T
   (2) Tnew ← T
   (3) foreach z ∈ V
   (4)   foreach (x, y) ∈ Fnew, (x, y) ∈ E, z ∉ S(x, Tnew)
   (5)     Ttest ← Tnew / {(x, y)}∪{(x, z)}
   (6)     if H(Ttest) ≤ H(Tnew)
   (7)       Tnew ← Ttest
   (8) return Tnew
```

Рис. 5. Ітераційний алгоритм, у якому кожна модифікація відразу приймається або відхиляється

```
   Задано. Граф G = (V, E). Маршрутне дерево T = (V, F, r), де
F ⊆ E. H(T) – сума відстаней між усіма невпорядкованими парами
вузлів у дереві T.
   Визначити. Маршрутне дерево Tnew, для якого H(Tnew) ≤ H(T).
   (1) Ttest ← T
   (2) Tnew ← T
(3)    foreach z ∈ V
(4)      foreach (x, y) ∈ Fnew, (x, y) ∈ E, z ∉ S(x, Tnew)
   (5)     Ttest ← Tnew / {(x, y)}∪{(x, z)}
(6)      if H(Ttest) ≤ H(Tnew)
   (7)     Tnew ← Ttest
(8)    return Tnew
```

Рис. 6. Ітераційний алгоритм, у якому одночасно перевіряються декілька модифікацій

Для вироблення рішення щодо прийняття або відхилення кожної модифікації дерева необхідно визначати зміну величини H. У мережі з

$b = |E| - |V| - 1$ розірванними зв'язками необхідно $O(Nb)$ розрахунків, що обчислювально є дуже витратним, якщо значення b є великим.

Нехай $d(x, y)$ – відстань між вузлами x та y уздовж маршруту, обраного методом АМ. Для двох множин вузлів X та Y нехай $D(X, Y)$ – сума довжин маршрутів, які з'єднують кожну пару вузлів x, y ($x \in X, y \in Y, x \neq y$):

$$D(X, Y) = \sum_{x \in X,\, y \in Y} d(x, y) - \frac{1}{2} \sum_{x, y \in Y \cap X} d(x, y). \quad (1)$$

Другий член цього виразу введено для компенсації подвійного підрахунку вузлів, які належать як X, так і Y.

Необхідна наявність можливості швидкого оновлення значення H, коли у піддереві з коренем в a, предком b та розірваним зв'язком з вузлом c розривається зв'язок з b і c стає предком (передбачається, що результатом цієї модифікації є повнозв'язне дерево).

Нехай $S(a, T)$ – множина вузлів у піддереві дерева T з коренем в a. Нехай $S'(a, T) = V \backslash S(a, T)$. Нехай T_1 – маршрутне дерево мережі. Для стислості запишемо $S_1 = S(a, T_1)$ и $S'_1 = S'(a, T_1)$. Нехай a – вузол з предком b в дереві T_1.

Тоді

$$H(T_1) = \frac{1}{2} \sum_{x, y \in V} d(x, y) = \frac{1}{2} \left[\sum_{x, y \in S_1} d(x, y) + 2 \sum_{x \in S_1,\, y \in S'_1} d(x, y) + \sum_{x, y \in S'_1} d(x, y) \right]. \quad (2)$$

Оскільки S_1 і S'_1 є неперетинними (непересічними) множинами, якщо $x \in S_1$ і $y \in S'_1$, тоді:

$$d(x, y) = d(x, a) + d(a, b) + d(b, y) = d(x, a) + 1 + d(b, y) \quad (3)$$

і

$$\sum_{x \in S_1, y \in S'_1} d(x, y) = N(S_1) N(S'_1) + \sum_{x \in S_1, y \in S'_1} \left(d(x, a) + d(b, x) \right) \quad (4)$$

$$= N(S_1) N(S'_1) + N(S'_1) \sum_{x \in S_1} d(x, a) + N(S_1) \sum_{y \in S'_1} d(b, y)$$

$$= N(S_1) N(S'_1) + N(S'_1) D(\{a\}, S_1) + N(S_1) D(\{b\}, S'_1)$$

Використовуючи вирази 1, 2 і 4, отримаємо:

$$H(T_1) = D(S_1, S_1) + D(S'_1, S'_1) + N(S_1) N(S'_1) + N(S'_1) D(\{a\}, S_1) + N(S_1) D(\{b\}, S'_1) \quad (5)$$

Нехай T_2 – така модифікація дерева T_1, що c є предком a, де $c \notin S_1$. Передбачається, що $S_2 = S(a, T_2) = S_1$. Позначимо функції відстаней мереж для дерев T_1 і T_2 як $H(T_1)$ і $H(T_2)$ відповідно. Тоді $H(T_2)$ може бути представлене аналогічно виразу 5:

$$H(T_2) = D(S_1, S_1) + D(S_1', S_1') + N(S_1)N(S_1') + N(S_1')D(\{a\}, S_1) + N(S_1)D(\{c\}, S_1').\quad (6)$$

Враховуючи 5 і 6, отримаємо:

$$\Delta H = H(T_2) - H(T_1) = N(S_1)\left[D(\{c\}, S_1') - D(\{b\}, S_1')\right], \quad (7)$$

де:

$$D(\{b\}, S_1') = D(S_1 \cup S_1', \{b\}) - D(S_1, a) - N(S_1),$$

$$D(\{c\}, S_1') = D(S_1 \cup S_1', \{c\}) - N(S_1)d(b, c) - N(S_1) - D(S_1, a).$$

Таким чином, отримаємо:

$$\Delta H = N(S_1)\left[D(V, c) - D(V, b) - N(S_1)d(b, c)\right]. \quad (8)$$

Практичні результати

На рис. 7–9 наведені результати застосування алгоритмів поліпшення на вихідному дереві, сформованому методом ПВШ. Для кожної мережі застосовувались три методи. Необхідно відзначити, що поліпшення H, описані нижче, будуть варіюватися від мережі до мережі. Отримані результати не є показовими для всіх мереж цього розміру, а лише на прикладі демонструють деяку типовість поведінки та характеристики масштабованості. Далі в цьому розділі будуть використані більш реалістичні мережі, отримані на основі знімків Інтернет та за допомогою сучасних засобів генерування мереж.

Перший метод (рис. 7а, 8а, 9а) відповідає описаному раніше алгоритму, наведеному на рис. 5. Провадиться інтеграція за кожним вузлом у дереві, яка робить цей вузол предком будь-яких сусідніх вузлів, з якими він зв'язаний розірваним зв'язком, потім модифікація негайно приймається або відхиляється. Загальне поліпшення залежить від вихідного дерева, яке на кожному кроці залежить від вибору кореневого вузла. H приймає мінімальне значення з випадковим локальним максимумом. Цей максимум відбувається, коли дві чи більше зміни разом викликають зниження H, а окремо спричиняють підвищення, а отже, відхиляються алгоритмом.

Другий метод (рис. 7б, 8б, 9б) відповідає описаному раніше алгоритму, наведеному на рис. 6. У цьому випадку вузол стає предком для всіх можливих сусідніх вузлів і загальна модифікація дерева приймається або відхиляється. Результати не дуже відрізняються від результатів першого методу. Локальний мінімум H все ще наявний.

Третій метод є більш цікавим. Він являє собою комбінацію двох перших методів. Спочатку застосовується перший метод, далі другий, а потім повторно перший метод. Результати, наведені на рис. 7в, 8в і 9в, помітно перевершують попередні – локальних мінімумів значно менше.

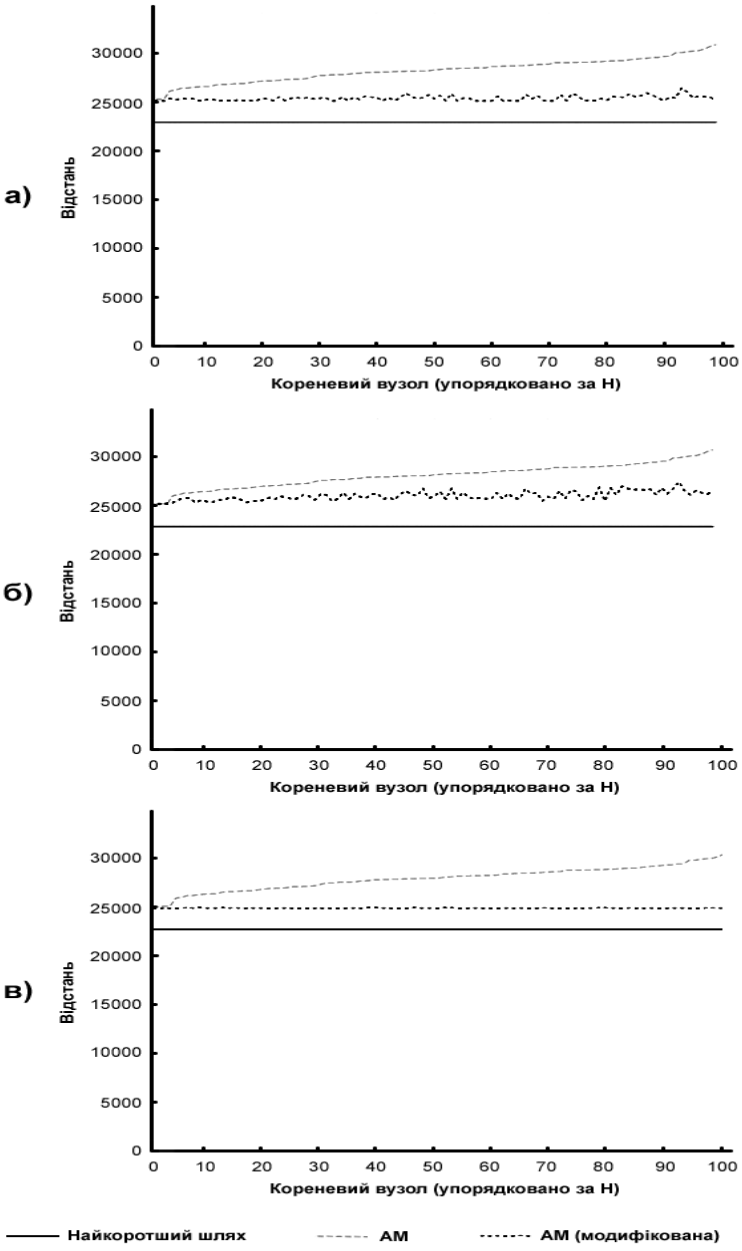


Рис. 7. Приклад поліпшення H для мережі, яка містить у собі 100 вузлів

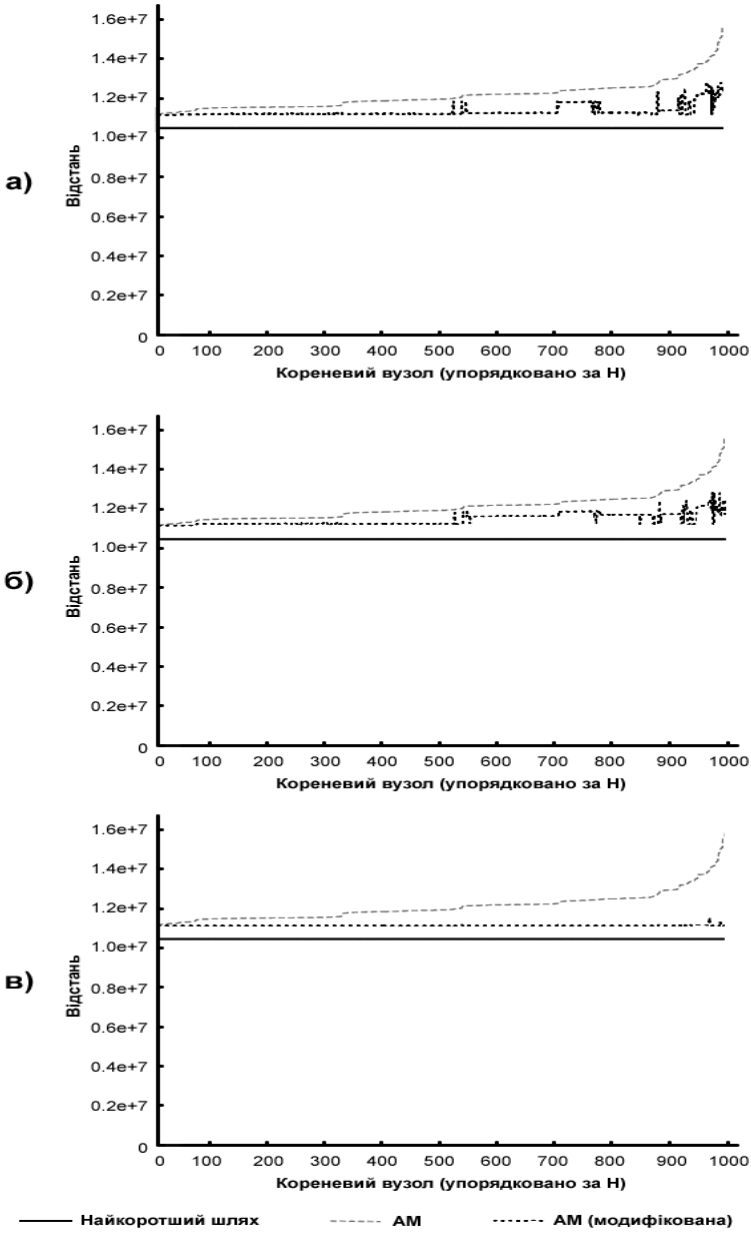


Рис. 8. Приклад поліпшення H для мережі, яка містить у собі 1500 вузлів

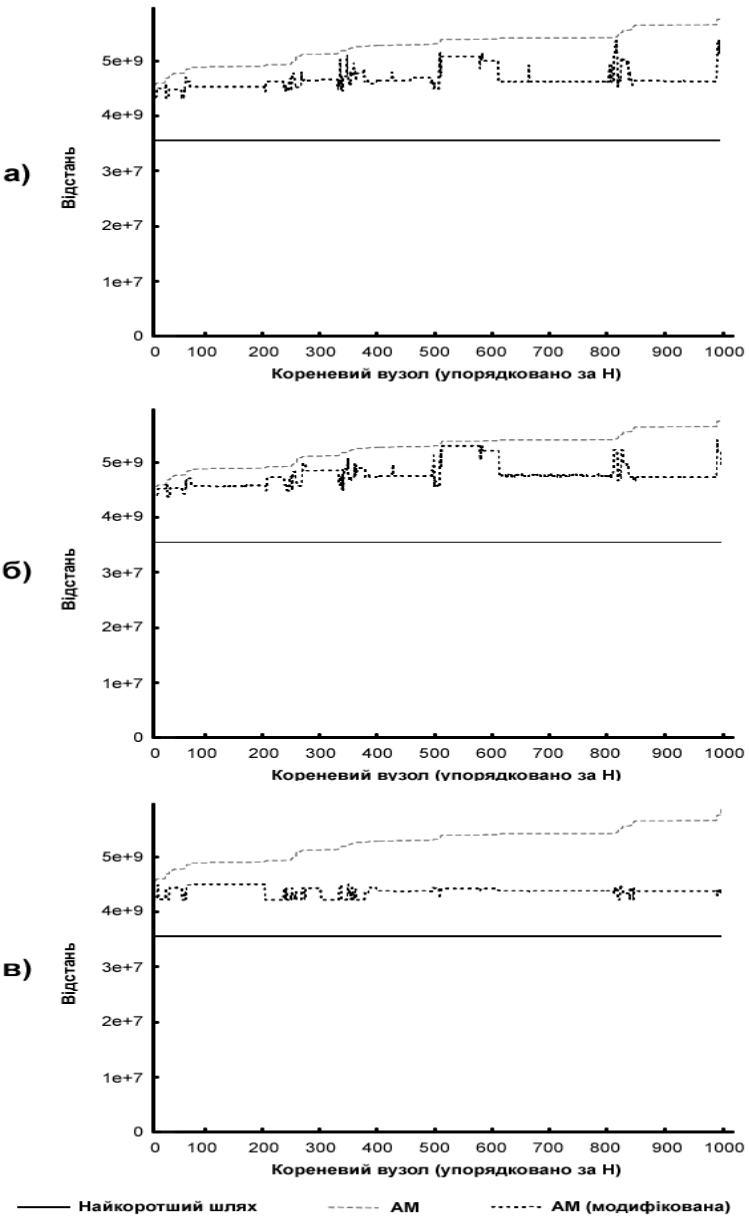


Рис. 9. Приклад поліпшення H для мережі, яка містить у собі 25000 вузлів

У середньому для розглянутих мереж алгоритми поліпшення знижують значення H у межах 6 ÷ 17 відсотків. Хоча таке поліпшення і не є настільки значним, його невелика обчислювальна складність виправдовує хоча б один прохід процесу поліпшення.

Час, необхідний для модифікації дерева за допомогою алгоритмів, описаних вище, суттєво не відрізнявся.

Рис. 10 ілюструє витрати часу на модифікацію дерев ПВШ мереж різних розмірів. Ці показники були отримані шляхом поодинокого вимірювання та ілюструють порядок величини часу, а не конкретне значення продуктивності. Для виконання тестів використовувався ПК на базі процесора AMD Athlon 1800МГц з 512Мб ОЗП під управлінням ОС FreeBSD 5.3 з компілятором GCC 3.4.2.

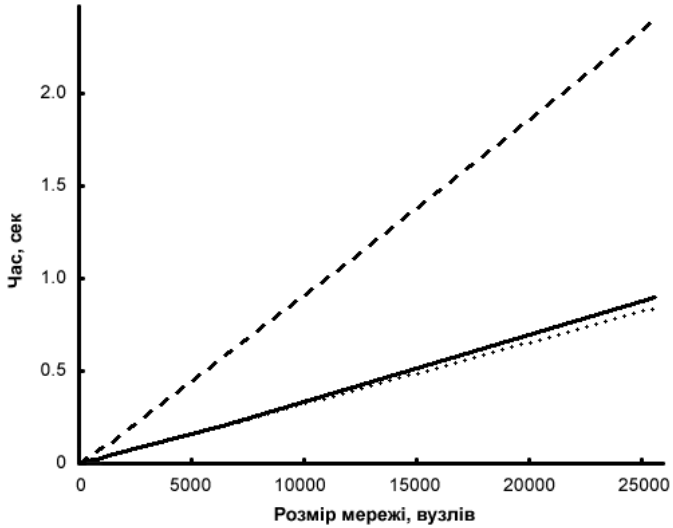


Рис. 10. Час, витрачений на поліпшення якості маршрутного дерева

Висновки

Важливою складовою моделювання обчислювальних мереж є моделювання механізмів маршрутизації. При виконанні моделювання мають бути збалансовані такі фактори: швидкість, точність і масштабованість. Найбільш точний метод передбачає докладне моделювання протоколів маршрутизації для кожного вузла мережі. Проте, якщо кожен з модельованих маршрутизаторів підтримує повні таблиці маршрутизації, значні вимоги щодо пам’яті перешкоджають моделюванню великих мереж. Для вирішення цієї проблеми у модель може бути введений певний рівень абстракції.

Метод АМ передбачає відображення мережі на дерево та використовує простий алгоритм розрахунку маршруту між двома заданими вузлами. При цьому отримані маршрути не завжди є найбільш короткими, у них можуть виникати деякі спотворення.

У цій статті запропоновані швидкі та ефективні методи підвищення якості маршрутних дерев. Модифікація дерев, які використовуються в АМ, дає можливість помітно скоротити довжину маршрутів, що отримуються, а це особливо важливо у завданнях моделювання надвеликих обчислювальних мереж. У статті продемонстровано результати практичного застосування описаних методів для модифікації маршрутних дерев мереж різних розмірів. Відносна простота методів дає змогу застосувати їх у випадках, коли використання АМ декількох дерев обчислювально не є виправданим.

Перелік посилань

1. *Huang P.* Enabling Large-scale Network Simulations: A Selective Abstraction Approach. PhD thesis, University of Southern California, 1999. – P. 129.
2. *Huang P., Heidemann J.* Minimizing routing state for light-weight network simulation // In Proceedings of the International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems, Cincinnati, Ohio, USA. - August 2001. – P. 108 – 116.
3. *Raman S., McCanne S., Shenker S.* Asymptotic scaling behavior of global recovery in SRM // In Proceedings of SIGMETRICS/PERFORMANCE 98, Joint International Conference on Measurement and Modeling of Computer Systems. - 1998. – P. 90 – 99.
4. *Корольков И. В.* Повышение производительности моделирования маршрутизации в больших вычислительных сетях // Радіоелектронні і комп'ютерні системі. – 2007 - №7(26). – С. 20 – 26.
5. *Hu T. C.* Optimum communication spanning trees // SIAM Journal of Computing. - 3(3). - 1974. - P. 188 – 195.
6. *Кормен Т., Лейзерсон Ч., Ривест Р.* Алгоритмы: построение и анализ. - М.: МЦНМО, 1999. – С. 1085 – 1151.
7. *Li Y., Bouchebaba Y.* A new genetic algorithm for the optimal communication spanning tree problem // In Artificial Evolution, volume 1829 of Lecture Notes in Computer Science. - 1999. – P. 162 – 173.
8. *Rothlauf F., Gerstacker J., Heinz A.* On the optimal communication spanning tree problem. IlliGAL Report No. 2003015. - May 2003. – P. 12