ёУДК 681.3

О.М. Данильченко, к.т.н., доц.
С.М. Защипас, аспір.
Житомирський інженерно-технологічний інститут

СТВОРЕННЯ BEOWULF-ПОДІБНИХ ОБЧИСЛЮВАЛЬНИХ КЛАСТЕРІВ

В статті розглядаються основні напрямки в області високопродуктивних обчислень. Описується структура розробленої авторами Beowulf-подібної кластерної системи, розрахованої на використання діючих комп'ютерних класів та мереж, що динамічно підключаються до кластера.

В даний час усе більшого розвитку набуває новий напрямок в області комп'ютерної індустрії – метакомп'ютерінг, головною рушійною силою якого виступають високопродуктивні пакети програм, що вимагають комп'ютерних архітектур із піковими на сьогоднішній день технічними характеристиками, які необхідні для вирішення значних для суспільства проблем науки та інженерії. Наведемо для прикладу декілька, найбільш розповсюджених застосувань задач такого класу:

– геоінформаційні системи;
– астрономія;
– обробка статистичних даних;
– обробка інформації з космічних метеостанцій;
– моделювання надзвичайних ситуацій у різних технічних системах;
– обробки цифрових зображень, фільмів, розпізнавання образів.

Такі задачі характеризуються наступними вимогами до комп'ютерних ресурсів: швидкодія 0,2–20×10^{12} операцій з плаваючою точкою в секунду; оперативна пам'ять 100–200 Гбайт; дискова пам'ять 1–2 Тбайт. Для їхнього рішення існує два основних підходи:

1. Створення суперкомп'ютерів.
2. Створення розподілених обчислювальних систем на базі внутрішніх комп'ютерних мереж і підключення їх через Internet до обчислювальної інфраструктури національного і світового масштабів.

Під терміном "суперкомп'ютер" звичайно розуміють спеціально виготовлений апаратно-програмний комплекс, в якому об'єднується велика кількість спеціально розроблених для нього надшвидких процесорів, надшвидка пам'ять та надшвидка система вводу–виводу. Процесори в суперкомп'ютері з'єднані між собою в складну багатовимірну ієрархічну структуру, яка дозволяє передавати дані до процесора з високою швидкістю. Таким чином, можливість вирішування складних задач на таких комп'ютерах досягається за рахунок паралельних обчислень та за рахунок використання спеціальної надшвидкої апаратури. На сьогоднішній день як приклад суперкомп'ютера можна навести унікальні архітектури – Intel Paragon, ABMSP, CRAY, Hitachi та інші. Подібні системи містять тисячі процесорів. З іншого боку, сумарний об'єм ресурсів в мережі уже зараз далеко перевищує ці показники; питання в тому, як об'єднати ці ресурси і передати в руки реальному користувачу [2, 3].

В 1994 році в NASA було створено апаратно-програмну систему із декількох десятків стандартних бездискових робочих станцій на базі процесорів Intel 80486, об'єднаних в мережу під управлінням операційної системи Linux. Система, що отримала назву Beowulf, була настільки вдалою, що стала індустріальним стандартом на створення обчислювальних кластерів [4].

Beowulf-подібна розподілена комп'ютерна система-кластер являє собою паралельну систему зв'язаних між собою комп'ютерів, доступних користувачу у вигляді єдиного обчислювального ресурсу. Технологічний прогрес в області високошвидкісних комп'ютерних мереж робить кластери або мережі робочих станцій привабливим рішенням для виконання дешевих та ефективних паралельних розрахунків. Сучасні акценти паралельних обчислень зміщуються зі спеціалізованих платформ на більш дешеві системи загального призначення, що широко використовуються в персональних комп'ютерах [3].

Пропоновані вітчизняним користувачам багатопроцесорні обчислювальні системи середнього класу характеризують вкрай невигідне співвідношення ціна–продуктивність через дорожнечу технічних рішень і високу вартість кваліфікованої робочої сили в західних країнах. Вибір базової платформи на дешевих

доступних комплектуючих для персональних комп'ютерів і збір паралельних систем у нас дозволяє обійти головні джерела збільшення собівартості та, власне, кінцевої вартості такої техніки.

Розподілена система-кластер ЖІТІ є Beowulf-подібною системою, яка являє собою апаратно-програмний комплекс, що відповідає таким характеристикам:

– підтримується робота в режимі on-line;

– комплекс має низьку вартість експлуатації та можливість максимальної автономності роботи комплексу;

– комплекс інтегровано в існуючу мережу інституту;

– існує мобільна масштабованість системи у рамках вже діючих комп'ютерних класів для проведення обчислень великого об'єму;

– підтримується необхідний рівень захисту та система аутентифікації користувачів системи;

– забезпечується контроль за процесом обчислень;

– існує підсистема оцінювання вартості обчислень;

– система має можливість надавати обчислювальні послуги та контролювати виконання обчислень.

Апаратно-програмний комплекс розподілених обчислень побудований на операційній системі Linux. В порівнянні з іншими операційними системами існує досить багато повнофункціональних та безкоштовних дистрибутивів Linux, які містять в собі всі необхідні засоби для організації розподілених систем:

– підтримка мережі;

– можливість віддаленого запуску програм на виконання інтегровано в операційну систему;

– безкоштовний компілятор мови С;

– безкоштовні засоби для організації MPI [5] (Message Passing Interface – інтерфейс, що базується на передачі повідомлень. Цей інтерфейс є одним із стандартних методів організації паралельних систем.);

– невибагливість до апаратних ресурсів;

– мультиплатформність;

– підтримка багатопроцесорності.

Ядро розподілених обчислень будується на кластері Linux машин, який побудовано таким чином.

В мережі існує ведучий вузол кластера системи. На жорсткому диску цього вузла встановлена операційна система Linux. Система встановлена таким чином, що жорсткий диск ведучого вузла можуть використовувати інші вузли; крім того, завантаження вузлових станцій також іде з ведучого вузла.

В результаті завантаження ведучого вузла та вузлових станцій в мережі існує декілька машин, на яких завантажена операційна система Linux. Ці станції можуть одночасно виконувати частини якоїсь загальної роботи.

Для цього програма сконструйована таким чином, що кожна окрема машина виконує обчислення якійсь певній області задачі. Виконувана програма розміщується на жорсткому диску ведучого вузла. Оскільки всі станції бачать один і той же жорсткий диск – диск ведучого вузла, то, використовуючи можливість Linux запускати на виконання завдання на віддалених комп'ютерах, створено програму, яка при виконанні на ведучому вузлі під'єднується послідовно до вузлових станцій та запускає на кожному з них на виконання частину задачі, а після вирішення всіх частин задачі об'єднує рішення.

Для розбиття задачі на частини та виконання частин на різних процесорах використовується бібліотека MPI, що підтримує багатопроцесорну обробку інформації. Можливо також безпосередньо виконувати завдання на віддалених машинах.

Середовище для виконання розподілених програм створено на базі стандартних бібліотек, що підтримують MPI.

Система складається з модулів, що надають їй необхідну функціональність.

Модуль для аутентифікації користувача

Цей модуль реалізує такі функції: додавання та вилучення користувача та надання йому привілеїв. Привілей грає роль у випадку, коли в чергу на вирішення задачі стоїть декілька користувачів. Тоді при складанні розкладу користувач з більш високим привілеєм отримає результат швидше за інших. Крім того, цей модуль виконує функцію прийому задачі користувача на виконання та відправки її процедурі, що складає розклад.

Модуль розкладу

Цей модуль складає розклад виконання задач згідно з привілеями та завантаженістю вузлів системи. Цей модуль оцінює ситуацію з завантаженістю системи та згідно з пріоритетом задач формує чергу задач.

Модуль черги задач

Цей модуль надає послуги щодо побудови та реорганізації черги задач. У тому випадку, якщо система вільна для виконання задачі, модуль передає першу задачу з черги модулю, що запускає задачі та автоматично реорганізовує чергу, зсуваючи покажчик початку черги на наступну задачу у черзі.

Модуль часу виконання

Модуль веде перевірку та компіляцію коду задачі, після чого передає її ядру системи для виконання. Крім того, цей модуль видає інформацію про завантаженість системи, часові виконання задачі, приблизний час до завершення виконання, ім'я власника задачі. За допомогою цього модуля можна призупинити чи завершити виконання задачі.

Модуль відображення

Модуль відображення – це інтерфейс користувача, який дозволяє візуально контролювати процес виконання задач. Цей модуль відповідає за інструменти відображення: відображення в текстовому режимі, відображення в графічному режимі, відображення у WEB навігаторах.

Модуль інтерфейсу роботи з користувачем

Модуль інтерфейсу роботи з користувачем надає користувачу інструменти для підключення до системи, формулювання задачі та контролю за її виконанням. Цей інтерфейс поєднує в собі управління всіма модулями, що описані вище.

Модуль автоматичної конфігурації системи

Модуль автоматичної конфігурації системи повинен містити в собі агенти, що слідкують за апаратним середовищем комплексу. У випадку підключення або відключення вузлів системи ці агенти автоматично налагоджують серверне оточення для правильного навантаження вузлів сервера та виключення помилок, пов'язаних з виключенням вузлів із системи.

Дистрибутив

Дистрибутив комплексу являє собою програму, за допомогою якої можна встановити пакет та автоматично налагодити ведучий вузол системи і створити дискети, із яких повинні завантажуватись вузлові станції.

Усі модулі, що наведені вище, складають єдиний пакет програм, який інтегрується в операційну систему Linux. Схему взаємозв'язків цих модулів зображено на рис. 1. Стрілки показують напрямки руху даних між модулями.

Для розробки та моделювання розподіленої системи було використано системні блоки комп'ютерів, що використовуються в навчальному процесі (комп'ютерні класи).

На даний момент система випробувалась на 10 робочих станціях Pentium III-700, на кожній з яких було встановлено оперативну пам'ять 64 Мб. Робота відбувалася в мережі зі швидкістю передачі даних 100 М/с.

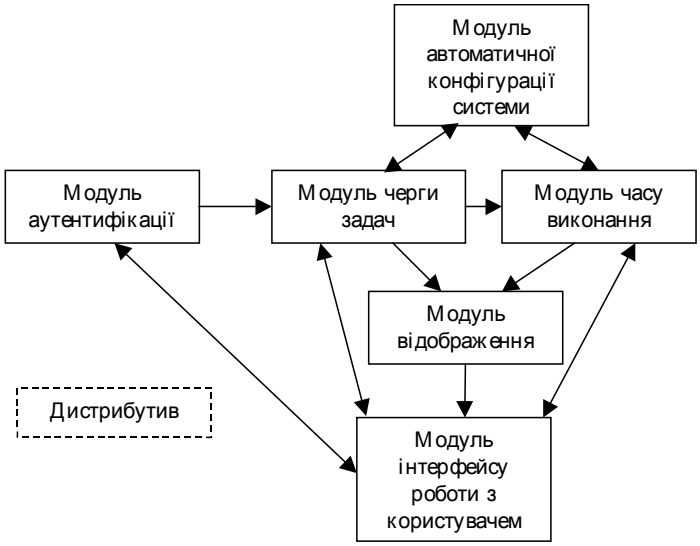

Рис. 1. Схема взаємозв'язків між модулями кластерної системи ЖІТІ

Серверним вузлом виступає робоча станція Pentium III-700, на якій встановлено 64 Мб оперативної пам'яті та жорсткий диск 7 Gb. Сервер працює під управлінням операційної системи Linux дистрибутива RedHat 7.0.

Схема розміщень робочих станцій зображена на рис. 2.

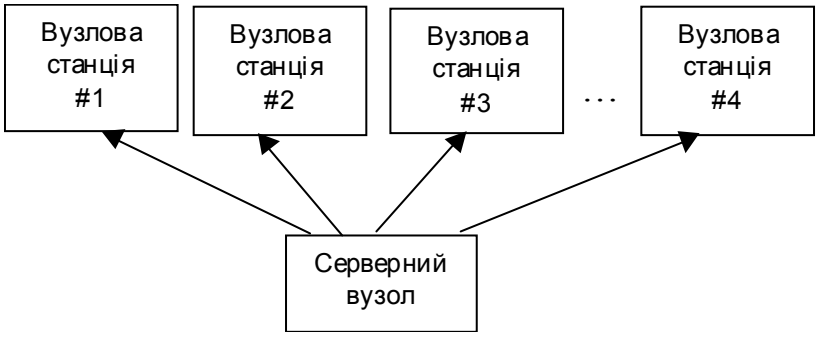

Рис. 2. Структурна схема кластера ЖІТІ

Тепер, коли ми маємо деяке представлення щодо структурної схеми системи, дозвольте нам звернути увагу на те, як ми можемо використовувати доступні ресурси машин у комп'ютерній лабораторії. Наприклад, припустимо, що ми хочемо обчислити суму квадратних коренів усіх цілих чисел між 1 і 1000000000. Ми пишемо просту програму, яка робить те, що нам потрібно. Для наочності наведемо використання цієї програми через стандартний інтерфейс Linux, причому для рішення використовується лише один вузол:

```
[serge@serge parallel]$ time  start.sh –n 1 ./sqrt 1 1000000000

Summ =21081851083600.55

real 8m0.017s
user 8m0.002s
sys 8m0.013s
```

Отже, наша програма рахувала 8 хв. Виникає питання: що ми можемо зробити, щоб прискорити час виконання роботи? Очевидно, що слід розбити роботу на частини і обраховувати їх паралельно на декількох комп'ютерах. Оскільки програма призначена для демонстрації, то вона була початково зроблена для роботи на декількох комп'ютерах. Кількість комп'ютерів задається параметром – n.

```
[serge@serge parallel]$ time  start.sh –n 10 ./sqrt 1 1000000000

Summ =21081851083600.55

real 0m51.029s
user 0m0.023s
sys 0m0.081s
```

Отриманий час – приблизно 55 секунд. Легко бачити, що на 10 процесорах ми отримали прискорення приблизно в 9,4 раза, що є досить високим показником для 10-кратного збільшення процесорів. Наведений приклад навряд чи зустрічається на практиці. Більшість реальних задач для розподілених систем мають більш складні алгоритми, і для їх реалізації необхідно використовувати різні спеціальні методи розпаралелювання [1, 2]. Велика кількість таких методів реалізується для системи розподілених обчислень ЖІТІ.

ЛІТЕРАТУРА:

1. *Воеводин В.В.* Математические модели и методы в параллельных процессах. – М.: Наука, 1986. – 296 с.
2. *Головкин Б.А.* Вычислительные системы с большим числом процессоров. – М.: Радио и связь, 1985. – 317 с.
3. *Андреев А., Воеводин Вл., Жуматий С.* Кластеры и суперкомпьютеры – близнецы или братья? – http://www.osp.ru/os/2000/05-06/009.htm
4. The Beowulf Project – http://www.beowulf.org
5. MPI:A Message-Passing Interface Standard –http://parallel.ru/docs/Parallel/mpi1.1/mpi-report.html

ДАНИЛЬЧЕНКО Олександр Михайлович – кандидат технічних наук, доцент, завідувач кафедри програмного забезпечення обчислювальної техніки Житомирського інженерно-технологічного інституту.

Наукові інтереси:
– теорія розкладів;
– теорія складності екстремальних задач;
– паралельні обчислення.

ЗАЩИПАС Сергій Миколайович – аспірант Житомирського інженерно-технологічного інституту.

Наукові інтереси:
– теорія розкладів;
– паралельні обчислення;
– операційні системи.

Подано 17.09.2001

Данильченко О.М., Защипас С.М. Створення Beowulf-подібних обчислювальних кластерів
Данильченко А.М., Защипас С.Н. Создание Beowulf-подобных вычислительных кластеров.
Danilchenko O.M., Zaschipas S.M. Creation of Âåîwulf-similar computing clusters

УДК 681.3

Создание Beowulf-подобных вычислительных кластеров / А.М. Данильченко С.Н. Защипас

В статье рассматриваются основные направления в области высокопроизводительных вычислений. Описывается структура разработанной авторами Beowulf-подобной кластерной системы, рассчитанной на использование действующих компьютерных классов и сетей, которые динамически подключаются к кластеру.

УДК 681.3

Creation of Âåîwulf-similar computing clusters / O.M. Danilchenko, S.M. Zaschipas

The article elucidates the main directions in the field of high-perforance computations. The authors developed a Beowulf-similar cluster system designed for use in working computer classes and networks dynamic registering new labs and networks connected to the cluster.