УДК 004.75

Є.В. ЛАНСЬКИХ, В.В. ГУЧОК

Київський національний університет технологій та дизайну

ДОСЛІДЖЕННЯ ПРОДУКТИВНОСТІ КЛАСТЕРНИХ СИСТЕМ

У цій роботі описані принципи побудови кластерних систем та основні методи тестування, а також основні вимоги до тестових програмних комплексів. Набір тестових програм дає змогу визначити продуктивність системи та визначити основні характеристики

Ключові слова: кластерні системи, програмні комплекси,продуктивність,тестування

Приблизно до середини 1990–х років основні напрямки розвитку комп'ютерних технологій були пов’язані з побудовою спеціалізованих багатопроцесорних систем із масових мікросхем. Одним з підходів, що сформувався – SMP (Symmetric Multi Processing), мав на меті об’єднання багатьох процесорів з використанням спільної пам’яті, що значною мірою полегшувало процес програмування. Зберегти швидкодію таких систем при збільшенні кількості вузлів до десятків, було практично неможливо. Даний підхід виявився дуже дорогим в апаратній реалізації.

Значно дешевшим і практично безкінечно масштабованим виявився спосіб MPP (Massively Parallel Processing), при якому незалежні спеціалізовані обчислювальні модулі об’єднувались спеціалізованими каналами зв’язку, при чому ці два способи застосовувались для створення спеціалізованих комп'ютерів, і не використовувалися в інших цілях. Ідея створення так званого кластера робочої станції фактично є розвитком методу MPP, адже логічно, що MPP-система практично не відрізняється від звичайної локальної мережі. Локальна мережа стандартних персональних комп’ютерів, при відповідному ПО, використовувалась як багатопроцесорний комп'ютер і започаткувала розвиток кластерних систем.

Об’єкти та методи дослідження

Кластери – це модульна багатопроцесорна система, створена на базі стандартних обчислювальних вузлів, сполучених високошвидкісним комунікаційним середовищем. За призначенням розрізняють відмовостійкі кластери; кластери з балансування навантаження; обчислювальні кластери; системи розподілених обчислень.

Під час розробки кластеру необхідно визначити масштабність системи, тобто залежність між збільшенням кількості вузлів в кластері та збільшенням швидкості розрахунків. В результаті, при зростанні числа вузлів буде спостерігатися збільшення продуктивності до певної межі, після якої, продуктивність почне падати. Це обумовлено латентністю мережі. Саме для цього і необхідно провести тестування.

Постановка завдання

Основним завданням цієї роботи є огляд існуючих методів та визначення найбільш ефективних для дослідження продуктивності кластерних систем.

Результати та їх обговорення

Кластерні технології стали логічним продовженням розвитку ідей, закладених в архітектурі MPP систем.

Якщо процесорний модуль в MPP системі є закінченою обчислювальною системою, то наступний крок – використання, у якості таких обчислювальних вузлів, звичайних персональних комп'ютерів.

Розвиток комунікаційних технологій, а саме, поява високошвидкісного мережевого обладнання та спеціального програмного забезпечення, такого як система MPI, реалізує механізм передачі повідомлень над стандартними мережевими протоколами, зробили кластерні технології загальнодоступними. Сьогодні не складає великих труднощів створити невелику кластерну систему, об'єднавши обчислювальні потужності комп'ютерів окремої лабораторії або навчального класу. Привабливою рисою кластерних технологій є те, що вони дозволяють для досягнення необхідної продуктивності об'єднувати в єдині обчислювальні системи комп'ютери самого різного типу, починаючи від персональних комп'ютерів і закінчуючи потужними суперкомп'ютерами.

Широке поширення кластерні технології отримали, як засіб створення систем суперкомп'ютерного класу із складових частин масового виробництва, що значно здешевлює вартість обчислювальної системи. Зокрема, одним з перших був реалізований проект COCOA, в якому на базі 25 двопроцесорних персональних комп'ютерів загальною вартістю порядку $ 100000 була створена система з продуктивністю, еквівалентній 48-процесорному Cray T3D, вартістю кілька мільйонів доларів США [1].

На даний момент можна констатувати, що створення і використання багатопроцесорних (паралельних) обчислювальних систем є найважливішою складовою розвитку комп'ютерної техніки.

У свою чергу, кластерні обчислювальні системи складають помітну частину паралельних обчислювальних систем і грають найважливішу роль в зв'язку з деякими властивими їм унікальними особливостями: відносно низькою вартістю, порівняльною простотою розгортання, можливістю поступового нарощування потужності при одночасному використанні «старого» обладнання тощо.

У зв'язку з цим, всі найбільші наукові центри та провідні промислові підприємства або володіють, або мають плани установки паралельних обчислювальних систем кластерного типу.

Таким чином, питання, пов'язані з проектуванням, побудовою і подальшим ефективним використанням кластерних систем, є вкрай актуальними. Одна з найважливіших проблем, яка виникає на всіх етапах життєвого циклу кластерної системи - від проектування до підтримки продуктивної експлуатації – проблема тестування кластерної системи [2].

Необхідність тестування

Перш ніж перейти до розгляду конкретних методик тестування і програмних продуктів, слід визначитися, чому взагалі виникає необхідність тестування кластерних систем. Виявляється, що підстав для проведення тестів може бути дві – визначення працездатності системи та виявлення істотних характеристик (особливостей) кластерної системи.

Перша підстава є абсолютно зрозумілою і навряд чи потребує якихось пояснень - кластер, як і будь-яка інша система, що експлуатується в промисловому режимі повинен мати засоби самодіагностики.

Такі засоби, як правило, спираються на засоби діагностики, вбудовані в операційну систему, мережеве обладнання або комунікаційне програмне забезпечення.

Слід зазначити, що побудова подібних систем є важливим завданням і дуже часто вона вирішується в рамках спеціалізованих програмних комплексів, названих системами управління кластерами (Cluster Management System, CMS). У рамках цієї статті будуть розглядатися тести другого роду – що виявляють деякі суттєві характеристики тестованих систем. Як істотні, ми будемо розглядати такі характеристики, які безпосередньо впливають на продуктивність систем. Вимірювання цих характеристик для конкретного кластера дозволяє вирішити відразу кілька важливих завдань:

- порівняти отриману обчислювальну систему з іншими аналогічними (що може дати деяке усереднене розуміння того, правильно налаштована система або виявити «вузькі місця» у обчислювальної системи, що тестувалася, усунення яких потенційно здатне підвищити загальну продуктивність;
- обрати алгоритм вирішення задачі (або його готову реалізацію), дозволяє максимально ефективно «підлаштуватися» під наявні характеристики обчислювальної системи;
- визначити на етапі тестування чи достатня потужність кластера для вирішення конкретного завдання в заданий час чи ні і т.д.

Крім зазначених, досить очевидних застосувань вимірювань, можна запропонувати і більш складні ситуації.

Скажімо, аналізуючи опубліковані результати тестів різних систем і володіючи інформацією про показники, яких необхідно досягти, можна на етапі проектування спробувати визначити необхідний програмно-апаратний склад майбутньої системи. Або, скажімо, знаючи для різних класів задач необхідні вимоги до відповідних характеристик, заздалегідь підібрати кластер, найбільш придатний для них. Ну і, нарешті, з'являється можливість оцінювати час виконання тієї чи іншої задачі (за умови наявності оцінки її складності, вираженої в одиницях вимірюваних тестом), що дозволяє реалізовувати різні варіанти планування виконання завдань на кластері.

Таким чином, результати тестів дозволяють:

- на стадії проектування ухвалювати рішення про склад майбутньої обчислювальної системи;
- на стадії налагодження і введення в експлуатацію шляхом порівняння з аналогами приймати рішення про досягнення необхідних характеристик;
- на стадії експлуатації (модифікації) виявляти вузькі місця системи, що робить тестування необхідною складовою всіх етапів процесу роботи з кластерною обчислювальною системою [2].
- Тестування абсолютно необхідне і його результати явно або опосередковано використовуються на всіх етапах життєвого циклу обчислювальної кластерної системи. Нижче, наведені деякі вимоги, яким повинні задовольняти тести, для того щоб вони могли бути використані для досягнення сформованих цілей:
- Повнота - тест повинен оцінювати тільки ті параметри, для оцінки яких створювався. Отримані результати повинні бути несуперечливими, лаконічними і легкими для розуміння.
- Легкість у використанні.

- Масштабованість. Тест повинен бути доступний для великої кількості різного за обчислювальною потужністю апаратного забезпечення.
- Переносимість. Тест повинен бути доступний для великої кількості різного по архітектурі апаратного забезпечення. Основною рисою переносимості є мова програмування, і, відповідно, наявність компілятора під дану платформу.
- Репрезентативність. Незалежно від платформи тест повинен завантажувати систему аналогічно використовуваними користувачами додатками. Результати подібних за структурою тестів повинні бути порівняні.
- Доступність. Тест повинен бути доступний, у тому числі і його вихідний код. Однак якщо тест поширюється разом зі своїми вихідними кодами, то при поданні результатів повинна бути вказана версія і всі внесені зміни.
- Відтворюваність. При необхідності повинна бути можливість повторити тест з отриманням аналогічних результатів. Для цього при публікації результатів необхідно надавати вичерпну інформацію про програмне і апаратне забезпечення.
- Всі тестові програми, що задовольняють зазначеним вимогам (а їх зараз існує безліч) можна класифікувати наступним чином:
- «Іграшкові» тести (toy benchmarks) - маленькі, довжиною в кількасот рядків вихідного коду. Як правило, такі тести являють собою рішення якої-небудь широко відомої математичної задачі – швидке або пірамідальне сортування, перемішування і т.д.
- Мікротести (microbenchmarks) – спеціалізовані, орієнтовані на визначення однієї з основних кількісних характеристик апаратного забезпечення - серед тестованих характеристик може бути:
- продуктивність центрального процесора;
- продуктивність і пропускна здатність локальної оперативної пам'яті;
- швидкість базових операцій введення/ виводу;
- продуктивність і пропускна здатність комунікаційного середовища.

В цю групу входять тести, що оцінюють продуктивність операцій, що вимагають синхронізації, і тести операційної системи (перемикання контекстів, системні виклики і створення процесів). Часто мікротести об'єднуються в пакети тестів.

- Ядра (kernels) - це фрагменти коду, взяті з реальних програм. Оскільки при виконанні програми проводять більшу частину часу саме в цих фрагментах, ядра дозволяють досить об'єктивно визначити швидкість виконання реальної програми на різних платформах.
- Синтетичні тести (synthetic benchmarks) оцінюють продуктивність на основі набору великої кількості показників і не прив'язані до якогось одного з додатків.
- Додатки (application benchmarks) - найбільш часто використовувані програми для реалізації тих чи інших реальних завдань. До цього ж класу можна віднести і

псевдододатки. Псевдододатки – це програми, створені на основі реальних додатків, але адаптовані з різних причин спеціально для задач тестування.

– Пакети тестів (benchmarks suites) – колекції різних типів тестів з перевагою додатків.

Щодо тестування кластерних систем, то зараз традиційно переважають тести, що відносяться до класу ядер (Linpack, NAS Parallel Benchmarks), додатків (NAS Parallel Benchmarks) і деяких мікротестів, як правило, для тестування комунікаційної складової (Netperf, тести лабораторії паралельних інформаційних технологій НДОЦ МДУ) [3].

Для тестування кластерних систем найчастіше використовуються два тести – Linpack Benchmark і NAS Parallel Benchmarks.

Linpack benchmark (LB) з'явився у 1979 році як додаток до бібліотеки Linpack (набір підпрограм для вирішення різних СЛАР). Метою створення LB було отримання можливості оцінки часу рішення тієї чи іншої системи рівнянь за допомогою пакета Linpack. Основним автором LB можна вважати J. Jack Dongarra. З тих пір Linpack був замінений новим і розширеним пакетом LAPACK, а LB залишився засобом для порівняння продуктивності комп'ютерів при роботі з плаваючою крапкою. З тих пір суть тесту не змінилася [4].

Суть цього тесту – рішення СЛАР Ax = f методом LU-факторизації (LU–розкладання) з вибором ведучого елемента стовпця. Де A - щільно заповнена матриця розмірності N.

Спочатку програма використовувалася для тестування окремих ЕОМ за допомогою матриці з N = 100. Із зростанням потужностей ЕОМ розмірність матриці була збільшена до N = 1000. Ці розмірності стали класичними і до цих пір застосовуються для тестування на FLOPS окремих машин.

Для задач тестування кластерів використовується зазначена вище версія тесту HPL. У цій версії користувач має можливість задати всі значущі параметри алгоритму, підбираючи їх для найкращої продуктивності.

Кожні півроку публікується звіт про продуктивність різних комп'ютерних систем на основі тесту LB – Linpack Benchmark report. Крім того, на сайті www.top500.org представлені 500 найшвидших комп'ютерів у світі за версією LB.

При паралельному процесі на обчислювальному кластері вихідна матриця поділяється на логічні блоки розмірністю NB x NB (зазвичай NB x NB при розрахунках лежить в інтервалі від 32 - 256). Ці блоки в свою чергу розбиваються сіткою PxQ на більш дрібні. Кожен з таких блоків «дістанеться» окремому процесору системи.

Коефіцієнти P і Q беруться в залежності від структури кластера, а їх добуток не може бути більше доступного числа вузлів.

Якщо в кластері 8 вузлів, то допустимими значеннями PxQ будуть: 1x8, 2x4, 3x2, 2x2, 1x4 ... При цьому в розрахунках братимуть участь PxQ процесорів.Саме процесорів, а не вузлів. Конкретні значення P і Q слід вибирати залежно від комунікаційного середовища.

За одну ітерацію головного циклу факторизації піддаються NB стовпців з наступним відновленням частини матриці. Результати розкладання пересилаються усіх вузлів одним з шести алгоритмів розповсюдження (broadcast algorithm).

Після розкладання послідовно вирішується дві системи рівнянь: Ly = f, Ux = y.

Тест вважається виконаним, якщо rn = O (1), r1 = O (1) і r ∞ = O (1), де rn=||Ax-f||∞ /(||A||1Nε); r1 =||Ax-f||∞ /(||A||1||x||1ε); r∞ =||Ax-f||∞ /(||A||∞||x||∞ε).

Ці умови означають, що задача вирішена правильно. Де ε - точність представлення чисел з плаваючою крапкою [5].

NPB з'явився на початку 90-х як дочірній проект при вирішенні агентством NASA задач обчислювальної гідродинаміки. Мета створення NPB була приблизно тією ж, що і у LB - оцінка необхідної потужності апаратного забезпечення для вирішення задач гідродинаміки. NPB спочатку був націлений на оцінку продуктивності паралельних обчислень на кластерах .

За задумом творців NPB є «paper and pencil» тестом, тобто офіційно NPB тільки набір правил і рекомендацій, доступних «на папері». Правила декларують практично всі питання, які можуть виникнути в процесі розробки:

- допустимі мови програмування;
- вичерпний опис всіх алгоритмів (ядер, додатків, генерації випадкових чисел ...);
- розрядності чисел з плаваючою комою;
- дозволу / заборони розпаралелювання деяких видів алгоритмів (наприклад, пошук мінімуму в векторі);
- введення / виведення;
- моменти вимірів часу, що включаються в тест операції;
- правила публікації результатів тестів;
- оцінку правильності отриманих результатів та інше.

Крім усього іншого на сервері NASA доступні готові реалізації тесту. У термінах розробників - це приклад коду (Sample Code), написаний на Fortran-77 і стандарті MPI. До версії 2.0 програма позиціонувалася як приклад реалізації, після версії 2.0 - це закінчений продукт.

Тест складається з низки простих синтетичних завдань: ядер (kernel benchmarks) і псевдо-додатків (application benchmarks), емулює обчислення на реальних завданнях (зокрема в галузі обчислювальної гідродинаміки).

У термінології NPB ядра і додатки можуть робити обчислення в певних класах завдань (Problem Classes): «Sample code», «Class A», «Class B» [1], «Class C» [2], «Class D» [3]. У NPB під класом розуміється розмірність основних масивів даних, що використовуються у тесті. Іншими словами, клас A - це маленькі матриці, B - великі, C - дуже великі, D - величезні.

Наприклад, для тесту на LU-розкладання, це буде розмірність вихідної матриці: 123, 643, 1023, 4083 для кожного з перерахованих вище класів відповідно. На поточний момент існує 5 ядер для визначення різних параметрів швидкодії паралельних систем з урахуванням певних критеріїв, а саме – EP, MG, CG, FT, IS.

Висновки

У даній статті розглянуто основні методи тестування кластерних систем, а також описані вимоги, яким повинні задовольняти тести, для того щоб вони могли бути використані. Було представлено опис двох тестів: Linpack Benchmark і NAS Parallel Benchmarks.

Linpack Benchmark дозволяє варіювати численними параметрами, за допомогою яких можна налаштувати і «поліпшити» кластер і порівняти з колосальною кількістю різноманітних систем.

Другий тест, навпаки, не дозволяє настільки вільно поводитися з параметрами розрахунку, а дозволяє виконувати обчислення тільки в заздалегідь визначених класах, також у своєму складі NPB має великий вибір засобів для визначення вузьких місць в обчислювальному процесі.

Список використаної літератури:

1. А. А. Букатов, В. Н. Дацюк, А. И. Жегуло Программирование многопроцессорных вычислительных систем. – Ростов-на-Дону. ООО «ЦВВР», 2003 – 23с.
2. http://winhpc.ru/?id=32
3. А.Н. Свистунов Технологии построения и использования кластерных систем. – Нижний Новгород, 2007. – 4 – 5с.
4. http://www.ixbt.com/cpu/cluster-benchtheory.shtml
5. Jack J. Dongarra, Piotr Luszczek, and Antoine Petitetz. The LINPACK Benchmark: Past, Present, and Future. December 2001.

Стаття надійшла до редакції 23.04.2012

Исследование производительности кластерных систем

Ланських Є.В., Гучок В.В.
Киевский национальный университет технологий и дизайна

Статья посвящена вопросу исследования производительности кластерных систем. Рассмотрены принципы построения кластерных систем и методы тестирования, а также основные требования до тестовых программных комплексов. Набор тестовых программ дает возможность определить производительность системы и основные характеристики.

Ключевые слова: кластерные системы, программные комплексы, производительность, тестирование.

Studies in productivity of claster systems

Lanskykh Y. V., Huchok V.V.
Kyiv national university of technology and design

The article is devoted to the problem of studying productivity of claster systems. Principles of structuring claster systems and principal methods of testing, as well as major demands to test program complexes are considered. A set of test programs allows to define productivity of programs the systems their main characteristics.

Keywords: cluster systems, software systems, performance, testing.