УДК 681.324

С.В. Мінухін, О.В. Мєзєнцев

Харківський національний економічний університет, Харків

ДОСЛІДЖЕННЯ МЕТОДІВ ЛОКАЛЬНИХ ПЛАНУВАЛЬНИКІВ РЕСУРСІВ ТА ЇХ МОДИФІКАЦІЇ В ГРІД-СИСТЕМАХ

Розглянуто дворівневу архітектуру планування ресурсів у Грід-системах. Проаналізовано локальні системи управління розподіленими обчисленнями та методи планування ресурсів. Проаналізовано алгоритми локального планування ресурсів на прикладі планувальника Maui. Наведена програмна база системи управління розподіленими обчисленнями. Досліджено поширення можливостей планувальника Maui шляхом інтеграції в програмне забезпечення нових алгоритмів планування ресурсів.

Ключові слова: *ресурси, Грід-система, система управління пакетного оброблення, PBS, кластер, Maui, локальний планувальник .*

Вступ

Концепція Грід зародилася в контексті важливої, але відносно вузької проблеми побудови надпотужних обчислювальних установок. Нове розуміння Грід широко трактується поняттям «ресурси» і включає в себе все, що бере участь в комп'ютерній обробці даних. У звичайній системі розподілених обчислень користувач може працювати тільки з тими ресурсами, де він зареєстрований, при цьому він повинен точно знати, де знаходяться його програми і дані. У Грід-системі користувач отримує доступ до ресурсів як спеціальний електронний сертифікат, а ця "розумна" система сама регулює пошук вільних ресурсів, звернення до сховищ даних в рамках своєї віртуальної організації. Сучасні Грід-системи вже зараз об'єднують значні ресурси, розташовані в різних наукових і технологічних центрах, інститутах, університетах світу. Вони включають окремі комп'ютери, кластери, сховища інформації, комунікації, програмні пакети та інструментарій [1].

Вихід за рамки завдань високопродуктивних обчислень виявляє реальний зміст Грід як інфраструктури для підтримки будь якої глобально розподіленого оброблення, для множини типів додатків: електронного бізнесу, розподіленого виробництва, дослідження даних, при цьому не потрібні високопродуктивні комунікації. Для розгляду всієї мережі як єдиного комп'ютера необхідно розглянути рішення задачі обліку в плануванні використання ресурсів не тільки географічної розподіленості ресурсів, але і їх тимчасової розподіленості.

Розподілені обчислення – це спосіб вирішення трудомістких обчислювальних завдань з використанням декількох комп'ютерів, найчастіше об'єднаних в паралельну обчислювальну систему. Виконання послідовних обчислень в розподілених системах має сенс в рамках вирішення багатьох завдань одночасно, наприклад в розподілених системах управління.

Мета роботи – дослідження та аналіз методів локального планування ресурсів та інтеграція нового методу планування в систему планування ресурсів.

Ієрархічна архітектура планування ресурсів у Грід-системах

Ефективність функціонування Грід-систем, як середовища з колективною формою обслуговування користувачів, визначається, в першу чергу, узгодженістю розподілу наявних ресурсів, яке має відбуватися з використанням процесів планування обчислювальних процесів в Грід в цілому. Однією з ключових функцій, необхідних для Грід-систем, є функція диспетчеризації, яка забезпечує розподіл ресурсів із загального ресурсного пулу між завданнями та доставку програм і даних для їх подальшого виконання [1].

В архітектурі Грід-систем ця функція реалізується програмними службами, що забезпечують такий рівень інтеграції розподілених ресурсів, при якому Грід-система представляється у вигляді єдиного операційного середовища оброблення запитів (завдань, програм, додатків користувачів). Більшість існуючих на сьогодні систем диспетчеризації призначено для обслуговування обчислювальних Грід-систем, що складаються з кластерів (кластеризованих ресурсів) – традиційної форми організації розподілених обчислювальних ресурсів в Грід-системах.

В якості базової архітектури досліджуваної системи в даній роботі обрана архітектура обчислювальної Грід-системи, що використовує ієрархічну структуру: на першому рівні знаходиться система планування ресурсів всієї Грід-системи (планувальник першого рівня), на другому – локальному рівні здійснюється локальне планування завдань на ресурси (локальний планувальник) , призначене для планування рішення призначених на даний ресурс завдань. Така система, відповідно до класифікації Грід-систем, називається дворівневою ієрархічною системою - з планувальником першого рівня і локальними планувальниками завдань на другому рівні. Для цього необхідно мати інформацію про вільні і доступні на моменти планування ресурси, їх кількості та характеристики, що дозволяють забезпечувати знаходження ресурсів (discovery resources), не-

обхідних для вирішення завдань користувачів, визначати такі, що найбільш підходять за своїми параметрами ресурси (matchmaking) в необхідні і обумовлені користувачами строки виконання їх завдань (директивні терміни).

Алгоритм оброблення завдань, що знаходяться в глобальній черзі, або черзі, що підготовлена для виконання завдань і яка склалася за певний проміжок часу, наступний: завдання обирається з глобальної черги і поміщується на ресурс (кластер, робочу стацію, тощо) за певним принципом (пріоритетом, вартістю, часом надходження і т.д.). Далі брокер першого (верхнього) рівня відповідно до існуючої політики обслуговування черги планує завдання на необхідні і доступні в даний момент часу обчислювальні ресурси. Після знаходження та вибору необхідного ресурсу завдання надходить в деякий проміжний пристрій для обслуговування обраного ресурсу і знаходиться в ньому до тих пір, поки цей пристрій не буде повністю заповнений. Після його заповнення завдання, що знаходяться в ньому, надходять на виконання на відповідні локальні ресурси. Описаний процес продовжується до тих пір, поки всі завдання з глобальної черги не будуть обрані, розміщені на ресурси та вирішені.

Таким чином, в системі функціонує два рівні управління завданнями глобальної черги – локальний і глобальний, кожен зі своїми об'єктами – завданнями, чергою, і системою управління – локальним монітором ресурсів (LRM) і Метадиспетчером (брокером першого рівня). Такий підхід реалізовано і досліджено в багатьох роботах і представляється перспективним з точки зору оптимізації організації проведення обчислень в умовах конкуруючих потоків і високої інтенсивності завдань користувачів [2].

Деякі питання планування ресурсів в Грід-системах

Планування в Грід розглядається як циклічний процес, який обробляє фіксовану на момент планування множину завдань, що знаходяться в черзі, і розподіляє їх по глобально розподіленим ресурсам, визначаючи час виділення ресурсів та їх адреси. Цей процес, виконуючись в контексті диспетчеризації, здійснює координацію поділу ресурсів між завданнями користувачів. У середовищі такого масштабу, як Грід-система, планування є найважливішим механізмом забезпечення якості обслуговування, яке полягає, насамперед, у забезпеченні прийнятного і передбачуваного часу виконання завдань користувача, а також гнучкості політики розподілу ресурсів у відповідності з пріоритетами [3].

Однак, відомі на сьогоднішній день проекти не здатні забезпечити необхідний для Грід рівень якості обслуговування і не дають вирішення завдання коалокації. Розглянемо, наприклад, European DataGrid (EDG) Resource Broker [4], який для кожного завдання відповідно до запиту користувача підбирає найбільш відповідний ресурс, ґрунтуючись на інформації, що надається інформаційною службою Globus MDS або R-GMA. Цією інформацією є відомості про статичні (кількість процесорів, операційна система, тощо) і динамічні (довжина черг) характеристики ресурсів.

Планування на основі такої інформації буде в ряді ситуацій призводити до затримки запуску завдань і навіть до їх зависання. Наприклад, нехай відомо, що на обчислювальному вузлі А черга містить 1 завдання, а на обчислювальному вузлі В черга містить 5 завдань, і розглядається завдання, якому по статичним параметрами підходять обидва цих вузла. На основі цієї інформації брокер розподілить завдання на вузол А. При цьому не можна передбачити, скільки це завдання буде чекати в черзі на обчислювальному вузлі А і коли буде виконано, тому що не приймається в розрахунок інформація про довжину, пріоритет та інші параметри завдання, яке знаходиться в черзі на обчислювальному вузлі Х. У цих умовах для вирішення проблеми коалокації ресурсів при плануванні багатопроцесорних завдань брокер був би змушений блокувати частину ресурсів на невизначений час, протягом якого вони будуть простоювати [3].

Однак EDG Resource Broker може дати добрі результати в спеціальних умовах застосування, наприклад, коли для користувача не має значення порядок виконання його завдань і час отримання результату, а використовувані ресурси повністю відчужуються від власників і централізовано управляються. Прикладами тут можуть служити різні обчислювальні завдання аналітичного характеру, як, наприклад, у проекті LCG, призначеному для обробки та аналізу експериментальних даних з Великого адронного колайдера [5]. Проте, з урахуванням колективного характеру функціонування Грід, ці умови не виконуються, і тому необхідно вирішувати завдання більш якісного планування, в тому числі, і багатопроцесорних завдань [3].

Ще один варіант реалізації планування в Грід був запропонований в Brigham Young University, де був розроблений метапланувальник Ursala [6]. Ця система використовує інтерфейс попереднього резервування, гарантує виділення ресурсів і запуск завдання в замовлений час, а також дистанційний інтерфейс з локальним планувальником Maui, що дозволяє визначити можливість виконання завдання на відповідному обчислювальному вузлі. Система також дозволяє планувати багатопроцесорні завдання, але для цього користувач повинен явно вказати ті ресурси, на яких він хоче розмістити своє завдання. З зазначених обчислювальних вузлів в систему надходять списки тимчасових інтервалів, на яких завдання може бути запущено, після чого планувальник визначає кращий час запуску завдання на цих ресурсах. Таким чином, основний недолік цієї системи полягає в тому, що вона здійснює планування,

яке вимагає від користувача конкретизації ресурсів для запуску завдання [3].

Система управління розподіленими обчисленнями PBS

Portable Batch System (PBS) – це система управління розподіленими обчисленнями. Вона являє собою набір комп'ютерів та інших ресурсів (мережі, системи зберігання даних, сервери, тощо). Деякі PBS складаються з всього лише декількох ПК, що виконують однопроцесорні обчислення. Управлінням такими системами займаються самі користувачі. Інші системи мають тисячі ПК, що виконують завдання користувачів системи, одночасно відстежуючи ліцензії на програмне забезпечення і доступ до обчислювальних ресурсів і систем зберігання даних. Менеджер ресурсів Torque здебільшого використовується саме в цих системах. Об'єднання ресурсів у PBS зазвичай зменшує технічне управління ресурсами, пропонуючи одноманітний підхід до користувачів. Після правильного налаштування, система абстрагується від багатьох деталей, пов'язаних з роботою та управлінням завданнями, що дозволяє підвищити рівень використання ресурсів. Наприклад, користувачам, як правило, потрібно вказати тільки мінімальні обмеження на завдання і не потрібно знати окремі імена машин кожного вузла (хоста), на якому запущене це завдання [7].

Portable Batch System складається з чотирьох різних компонентів:

1) головний вузол;
2) інтерактивні вузли;
3) обчислювальні вузли;
4) ресурси.

Головний вузол. PBS має головний вузол, де працює pbs_server (Portable Batch System Server). В залежності від потреб системи ресурси головного вузла можуть бути спрямовані на виконання поставленого завдання. Так само він може виконувати роль інших компонентів.

Інтерактивні вузли. Забезпечують точку входу в систему для користувачів для того, щоб вони мали можливість управляти робочим навантаженням цих вузлів. Для цих вузлів користувачі зможуть подавати заявку на виконання завдання, й так само відстежувати їхню роботу. Крім того, деякі сайти мають один або декілька вузлів, які зарезервовані для інтерактивного використання, наприклад, для тестування та усунення проблем. Ці вузли мають такі клієнтські команди, як qsub і qhold.

Обчислювальні вузли. Ці вузли є «робочими конячками» системи. Їх роль полягає у виконанні завдань, що надійшли в систему. На кожному обчислювальному вузлі працює демон (аналог служби у ОС Windows та ОС класу UNIX) pbs_mom, для старту, зупинки поставлених завдань, а також управління ними. Він взаємодіє з pbs_server на головному вузлі [7].

Ресурси. Деякі системи організовані спеціально для цілей управління ресурсами за межами обчислювальних вузлів. Ресурси можуть охоплювати високошвидкісні мережі, системи зберігання даних і так далі. Наявність цих ресурсів обмежена, і потрібно управляти ними розсудливо, щоб сприяти стійкості системи і більш широкого їх використання

Програмна база системи управління розподіленими обчисленнями PBS

В якості базового програмного забезпечення системи в даній роботі обрані такі програмні продукти, як Torque, Maui та операційна система сімейства Unix – CentOS.

Torque – це менеджер ресурсів, який є однією з версій Portable Batch System, що відповідає за відстеження доступної кількості ресурсів на вузлах кластера та запуск завдань. Torque управляє завантаженням обчислювальних комплексів, що складаються з певної кількості обчислювальних вузлів, що працюють під управлінням операційної системи сімейства Unix. Torque має вбудований планувальник завдань, який має назву pbs_sched, що визначає момент запуску завдань. Менеджер ресурсів забезпечує низькорівневі функціональні можливості такі, як запуск, утримування (тимчасове припинення), скасування і контроль виконання завдання. Без цих можливостей менеджера ресурсів планувальник не зможе самостійно контролювати виконання завдання. Torque кластер складається з одного головного вузла та багатьох обчислювальних вузлів. На головному вузлі працює демон pbs_server, а на обчислювальних вузлах запускається демон pbs_mom. Клієнтські команди для подання та управління завданнями можуть бути встановлені на будь-якому комп'ютері (в тому числі й на хостах, де відсутні демони pbs_server або pbs_mom) [8].

На головному вузлі також працює і демон планувальника. Планувальник взаємодіє з pbs_server для прийняття рішень, пов'язаних з локальними політиками щодо використання ресурсів і виділення вузлів для виконання завдань. Простий планувальник завдань FIFO, а також код для побудови більш складних планувальників наданий у дистрибутиві вихідного коду (source code). У більшості випадків користувачі Torque використовують планувальники з більш широким функціоналом, такі, як Maui або Moab.

Користувачі відправляють завдання на pbs_ server за допомогою команди qsub. Коли pbs_server отримує нове завдання, вона інформує про це планувальник. Коли планувальник знаходить вільні вузли для виконання завдання, він посилає інструкції для виконання завдання на вузлі зі списку вузлів pbs_server. Потім, pbs_server посилає нове завдання на перший вузол у списку вузлів і дає йому інструкції з запуску завдання. Цей вузол визначається як виконавчий хост і зветься Mother Superior. Інші вузли в ході виконання завдання називаються Sister Moms [8].

Maui – планувальник завдань, з розширеним функціоналом у паралельних обчислювальних системах. Він опитує Torque на предмет наявності вільних ресурсів і завдань у черзі, які необхідно виконати. На основі отриманих даних і своїх налаштувань він приймає рішення про запуск якого-небудь завдання і посилає команду серверу Torque виконати її. Maui дозволяє гнучко налаштовувати різні стратегії заповнення кластера, пріоритети для завдань за різними критеріями: пріоритетів, кількістю запитуваних ресурсів, тощо [9]. Як і для всіх інших планувальників, що застосовуються в системах пакетного оброблення даних (СПО), функції Maui наступні: при заданій множині готових до виконання завдань (з черги) і для поточного стану ресурсів визначається черговість запуску завдань та знаходяться відповідні ресурси для тих з них, які можна запустити в даний момент. У цих рамках Maui реалізує нові механізми, засновані на "передбаченні майбутнього" і спрямовані на оптимізацію управління виконання. Крім того, в Maui є ряд нововведень, що відрізняють його від попередників:

1) алгоритми зворотного заповнення (backfill) і справедливого розподілу ресурсів (fairshare) для підвищення ефективності системи та зменшення часу очікування в черзі;

2) система автоматичного визначення пріоритетів завдань, що дозволяє давати ключовим користувачам перевагу при розподілі ресурсів;

3) покращена діагностика проблем з завданнями, вузлами і програмними компонентами СПО;

4) розширений спектр статистики. За допомогою Maui адміністратор може отримувати повну історичну (хронологічну) та поточну інформацію про завдання, черги, планувальника, стан системи і т.п.

5) в Maui є можливість моделювання роботи СПО, за допомогою якої можна перевірити налаштування планувальника "у дії". Це дозволяє підібрати конфігурацію системи, яка найбільш повно відповідає вимогам конкретної виробничої обстановки [10].

У Maui є ще одна істотна відмінність: звичайні планувальники не мають власних командних інтерфейсів внаслідок того, що вони їм не потрібні. Maui ж, повністю реалізуючи функції резервування, має в своєму розпорядженні і відповідний інтерфейс для зовнішнього або, так званого, адміністративного резервування.

Механізм резервування Maui

Поняття резервування формалізується в Maui відповідним типом об'єктів.

Об'єкт "резервування" складається з:

1) ACL (Access Control List, список контролю доступу) – набору параметрів, керуючись яким планувальник визначає, для яких завдань доступні зарезервовані ресурси;

2) списку зарезервованих ресурсів;

3) часу дії резервування.

На рівні реалізації кожне резервування фізично являє собою запис у базі даних Maui. Об'єкт резервування прив'язаний, з одного боку, до часу, причому до майбутнього. З іншого боку, інтерпретація резервувань передбачає розгляд станів ресурсів на моменти початку резервувань. Зображується це у вигляді тимчасової розгортки (time line) станів ресурсів: для кожного ресурсу на вісі часу ставляться мітки, що визначають, яким завданням в цей момент ресурси можуть бути доступні. Сукупність певної кількості міток і утворюють одне резервування (тому зарезервувати можна цілий комплекс ресурсів, і всі вони будуть належати одному резервування) [10].

Зауважимо, що попереднє резервування проводиться під завдання, якого ще немає в СПО. Отже повинен бути механізм, який дозволяє зв'язати завдання зі зробленим резервуванням. Об'єкт резервування має список контролю доступу ACL, і саме відповідно з ним відбувається прив'язка завдання до певного резервування. Зарезервовані ресурси можуть отримати тільки такі завдання, які мають хоча б один з параметрів: GROUPLIST, USERLIST, ACCOUNTLIST, CLASSLIST і QOSLIST, – збігається за значенням з відповідними атрибутами ACL резервування. Тобто, прив'язка може здійснюватися за реєстраційним іменем користувача, його групі, класу і якості обслуговування. Можливості налаштування списків доступу ACL в Maui досить великі і, зокрема, можна домогтися того, щоб зарезервовані ресурси були доступні тільки для певного завдання, причому навіть ще не надійшовши у чергу СПО (саме це потрібно для метадиспетчеризації) [10].

Maui дозволяє зарезервувати 4 найбільш важливих типи ресурсів:

1) [PROCS = <INTEGER>:];
2) [MEM = <INTEGER>:];
3) [DISK = <INTEGER>:];
4) [SWAP = <INTEGER>:].

Кількість ресурсів визначається певним завданням – роботою. В роботу може входити кілька паралельних завдань, кожне з яких виконується на одному вузлі. Не слід плутати "роботи" та "завдання" (в документації Maui використовуються відповідно "job" і "task"). Кожна робота може складатися з кількох завдань. Завдання може виконуватися тільки на одному вузлі та є, як би, квантом роботи. Коли здійснюється резервування під роботу, то в запиті вказуються ресурси, необхідні для однієї роботи, і кількість завдань. Таким чином, всі завдання видаються однотипними, принаймні, які вимагають однакову кількість ресурсів. Далі розглядатимемо ті роботи, які мають тільки одне завдання.

В Maui передбачені засоби для операцій з існуючими резервуваннями. За допомогою команди «showres» можна отримати інформацію про те, де, для яких завдань, на який час і скільки ресурсів зарезервовано, причому доступний пошук, як по резерву-

ванню, так і по вузлах. Існує також команда «releaseres» для скасування створених резервувань [10].

Алгоритм планування Maui

Як і багато інших планувальники, Maui працює ітераційно, тобто перемежовуючи процес планування з очікуванням або виконанням зовнішніх команд. Кожен цикл починається при здійсненні однієї з таких подій:

1) змінюється стан завдання або ресурсу;
2) досягнуто межу резервування;
3) отримана зовнішня команда;
4) з початку попереднього циклу минув час, визначений як максимальний.

В процесі планування застосовується ще один, внутрішній тип резервування Maui – job-резервування. З точки зору реалізації він мало відрізняється від описаного вище адміністративного резервування.

Воно використовується в двох випадках:

- для забезпечення доступності ресурсів під виконуються завдання. Як тільки завдання запускається, то відразу, на весь час виконання (walltime), створюється job-резервування, що закриває доступ до ресурсів, необхідним для цього завдання, – всім, крім нього самого. Може статися, що під час роботи всі ці ресурси і не будуть використовуватися, однак як тільки вони будуть потрібні завданням, вони гарантовано зможуть їх отримати. Коли завдання завершується (можливо, раніше, ніж через час walltime) job-резервування скасовується, і ресурси стають вільними [10];

- job-резервування застосовується для того, щоб завдання з більш високими пріоритетами не були затримані запуском менш пріоритетних завдань, що може трапитися при роботі алгоритму backfill. В загальному вигляді процес запуску завдань виглядає так: до тих пір, поки це можливо, впорядковані за пріоритетами завдання беруться з черги і запускаються (попутно створюється job-резервування); як тільки чергове завдання не можна запустити через брак ресурсів для нього, визначається найближчим часом , в яке можна буде провести запуск, і починаючи з нього створюється job резервування затребуваних ресурсів на час walltime.

Після того як зроблена деяка, сконфігурована кількість резервувань, починає роботу власне алгоритм backfill. Він з'ясовує, які вузли і на який час, починаючи з поточного, вільні (це як раз визначається зробленим job-резервуванням). Після цього вільні вузли об'єднуються у «вікна». Сумарна «ширина» «вікон» може виявитися більше кількості вільних в даний момент вузлів, тому що деякі вузли можуть входити в кілька вікон. Потім з усіх вікон вибирається одне, як правило, саме широке. Серед усієї решти завдань вибирається і запускається те завдання, яке найбільш точно задовольняє цьому «вікну». Якщо є можливість, то запускається не одне завдання. Так як при запуску завдань алгоритмом backfill враховуються зроблені job-резервування, то відповідні їм завдання не будуть затримані [10].

Основною відмінністю job-резервування від адміністративного є те, що воно здійснюється самим планувальником і ним же відміняється перед початком чергового кроку планування. Причому картина резервувань може відрізнятися від попереднього кроку планування досить істотно. Це має місце у випадку, коли надходять нові завдання, що міняють порядок в черзі або звільняють ресурси. На відміну від адміністративних, job-резервування не гарантує запуск завдання в зазначений час. Таким чином, job- резервування (не для запущених завдань) є всього лише мітками, що дозволяють коректно врахувати в алгоритмі backfill пріоритети завдань. Адміністративні та job-резервування існують паралельно, тобто job-резервування можуть накладатися на адміністративні, якщо звичайне завдання, під яке робиться job-резервування, задовольняє ACL адміністративного [10].

Особливості планування багатопроцесорних завдань у Грід

Планування багатопроцесорних завдань у Грід пов'язане з низкою інших проблем, специфічних саме для Грід, які на наш погляд зводяться до наступних:

1) наявність гетерогенних ресурсів, тобто наявність глобально розподілених обчислювальних вузлів з різними платформами та характеристиками (кількість і продуктивність процесорів, об'єм оперативної пам'яті, об'єм фізичної пам'яті, тощо);
2) необхідність оптимального планування, яке полягає в тому, щоб для кожного завдання з черги підібрати не просто ресурси, що задовольняють ресурсному запиту, а такі, які є кращими за заданим користувачем критерієм оптимальності серед усіх глобально розподілених ресурсів;
3) необхідність здійснення синхронізованої за часом доставки інформації про стан ресурсів і прогнозу їх використання [3].

Аналіз роботи базового алгоритму планування Torque (FIFO) та алгоритму планування Maui (Backfill)

Розглянемо планування на основі пріоритетів, яке передбачає, що ресурси для більш пріоритетних завдань виділяються раніше, ніж для менш пріоритетних. В цих умовах широко використовуються алгоритми типу FCFS (FIFO), які працюють за принципом виділення звільнилися ресурсів самому пріоритетному завданню з черги, яке може на них розміститися. Цей тип планування використовує базовий планувальник Torque(pbs_sched). При цьому виходить, що більша частина процесорів буде завжди зайнята дрібними завданнями, тобто виникає

фрагментація ресурсів. Навіть якщо завдання має найвищий пріоритет, необхідний йому обсяг ресурсів може ніколи не утворитися і, отже, завдання може ніколи не стартувати. Для середовища, яке обслуговує однопроцесорні завдання, цієї проблеми немає. Вона виникає, коли є колективні ресурси, наприклад, при обслуговуванні багатопроцесорних завдань. Аналогічна ситуація виникає на машинах із загальною пам'яттю, в середовищі з загальним файловим простором та інших випадках, коли ресурси діляться між завданнями, а не виділяються під завдання цілком.

Для вирішення проблеми коалокації в локальних системах, наприклад, в планувальнику Maui, використовується алгоритм зворотного заповнення Backfill, розроблений для великих багатопроцесорних систем (MPP) типу IBM SP2.

Він має такі переваги:

1) в умовах роботи в пріоритетною системі дозволяє уникнути зависання завдання, гарантуючи його запуск;

2) ефективно завантажує ресурси, дозволяючи уникнути їх фрагментації;

3) має прийнятні часові характеристики при роботі на великій кількості обчислювальних вузлів;

4) дозволяє працювати на множині гетерогенних ресурсів.

Алгоритм зворотного заповнення Backfill працює за наступним принципом: розміщуючи найбільш пріоритетне завдання, він визначає момент часу, коли звільниться достатня кількість ресурсів, зайнятих завданнями, які вже виконуються і виконує резервування цих ресурсів. Завдання з меншим пріоритетом може бути запущено поза чергою, але тільки в тому випадку, якщо воно не буде заважати запуску всіх (в консервативному варіанті Backfill) більш пріоритетних завдань. Відзначимо, що в опублікованих роботах за алгоритмом Backfill міститься лише згаданий вище принцип, а також загальна схема процесу розподілу завдань з черги по ресурсах. Тому при адаптації цього алгоритму для Грід нам довелося займатися розробкою моделі даних для представлення інформації про стан обчислювальних вузлів і прогнозі їх використання, а також алгоритмом підбору ресурсів для завдань.

Однак варто зауважити, що алгоритм Backfill не є ідеальним і також має свої недоліки. Головним з них є те, що коли всі завдання вже розподілені на ресурси кластеру і виконуються, нові завдання, які надходять до черги у Грід не розподіляються за рівномірним законом(на перший ресурс, якій звільнився) на ресурси кластеру, а потрапляють на перший ресурс зі всього списку ресурсів, тобто у його початок. Цей недолік може зменшити продуктивність та швидкодію виконання завдань. Можливим рішенням цієї проблеми може бути доопрацювання планувальника Maui шляхом впровадження в нього додаткового алгоритму планування, який би забезпечив усунення цього недоліку. Проблема лише в тому, що розроблення власного алгоритму планування – задача складна та трудомістка та потребує трудових, часових та матеріальних ресурсів. Також це рішення потребує втручання у програмну середу планувальника Maui. Хоче з іншого, якщо алгоритм себе виправдає, то це може бути дуже вигідно[3].

Інтеграція нових алгоритмів в локальний планувальник Maui

Проаналізувавши каталог дистрибутиву Maui, можна зробити висновок, що є можливість впровадити власний алгоритм планування ресурсів. Така можливість надається у файлах коду дистрибутиву, а саме у:

\maui-3--*\src\moab\Mlocal,
\maui-3--*\src\moab\MQueue
\maui-3--*\src\moab\MSched

Згідно з коментаріями до коду, можна зрозуміти, де потрібно вставити код виклику та власне код самого алгоритму. Код алгоритму слід описувати у файлі вихідного коду «*\maui-3-*-*\src\moab\MQueue». У цьому файлі також присутній код алгоритму Backfill (функція «MQueueBackFill»). У файлі «*\maui-3-*-*\src\moab\MSched» є функція розподілу завдань («MJobDistributeTasks»), яка викликає та працює з функцією MQueueBackFill. У цьому файлі можна змінити функцію алгоритму BackFill на власну функцію з власним алгоритмом.

В такому випадку планування завдань буде автоматично проводитись по власному алгоритму. Також є можливість виклику власного алгоритму у коді файлу «*\maui-3-*-*\src\moab\Mlocal»:

```
int MLocalQueueScheduleIJobs(int    *Q,
mpar_t *P)
{
    mjob_t *J;
    int    jindex;
    if ((Q == NULL) || (P == NULL))
    {
            return(FAILURE);
    }
    /*NOTE:insert call to scheduling algorithm
    here*/
    for (jindex = 0;Q[jindex] != -1;jindex++)
    {
            J = MJob[Q[jindex]];
            /* NYI */
            DBG(7,fSCHED)       DPrint("INFO:
            checking job '%s'\n",
            J->Name);
```

```
        }
        /* END for (jindex) */
        return(FAILURE);
    }
     /* END MLocalQueueScheduleIJobs() */[9]
```

Виклик функції з власним алгоритмом треба робити в місці, де розробники залиши коментар: «NOTE: insert call to scheduling algorithm here».

Висновки

Таким чином, була розглянута дворівнева модель планування ресурсів у Грід-системах, а саме ієрархічна структура, на першому рівні якої знаходиться система планування ресурсів всієї Грід-системи (планувальник першого рівня), на другому, локальному рівні здійснюється локальне планування завдань на ресурси (локальний планувальник) , який призначений для планування рішення призначених на даний ресурс завдань. функцію диспетчеризації ресурсів, яка забезпечує розподіл ресурсів із загального ресурсного пулу між завданнями, доставку програм і даних для їх подальшого виконання.

Розглянуто програмну базу побудови системи управління розподіленими обчисленнями, а саме такі програмні продукти, як Torque та Maui, їх структура та загальні положення щодо механізмів їх роботи. Також був розглянутий алгоритм резервування та планування Maui.

Були висвітлені проблеми, пов'язані зі специфікою планування багатопроцесорних завдань у Грід, а саме проблеми гетерогенності ресурсів, необхідності оптимального планування та інші.

Проаналізовано особливості базових алгоритмів планування Torque та Maui. Приведена можливість включення нових алгоритмів планування у планувальник завдань Maui, що дасть можливість підвищити продуктивність роботи Грід-систем.

Список літератури

1. Пономаренко В.С. Методы и модели планирования ресурсов в GRID-системах: монография / В.С. Пономаренко, С.В. Листровой, С.В. Минухин, С.В. Знахур. – Харьков: ИД «ИНЖЭК», 2008. – 408 с.

2. Коваленко В.Н., Корягин Д.А. Полигон Грид в ИПМ РАН и разработка методов управления ресурсами в глобальной среде // X конференция представителей региональных научно-образовательных сетей, RELARN-2003, Санкт-Петербург, 16-20 июня 2003 // Сборник тезисов докладов, Тровант, 2003. – С. 214–216.

3. Коваленко В.Н., Семячкин Д.А. Использование алгоритма BACKFILL в ГРИД [Електронний ресурс]. – Режим доступу до ресурсу: http://gridclub.ru/library/publication.2004-12-27.6965428621/publ_file/.

4. The DataGrid Projec. [Електронний ресурс]. – Режим доступу до ресурсу: http://eu-datagrid.web.cern.ch/.

5. *LHC [Електронний ресурс]. – Режим доступу до ресурсу: http://lhc.web.cern.ch/lhc/*

6. Job scheduling strategies for parallel processing: IPDPS 2000 workshop, JSSPP 2000, Cancun, Mexico, May 2000, Proceedings[Електронний ресурс]. – Режим доступу до ресурсу: http://www.informatik.uni-trier.de/~ley/db/conf/jsspp/jsspp2000.html

7. Adaptive Computing – Documentation / TORQUE Administrator's Guide. Version 3.0.3 [Електронний ресурс]. – Режим доступу до ресурсу: http://www.adaptivecomputing.com/resources/docs/torque/3-0-3/index.php

8. Система пакетной обработки заданий Torque: Руководство пользователя. Т – Платформы, 2008. [Електронний ресурс]. – Режим доступу до ресурсу: hpc.ssau.ru/files/doc/torque_manual.pdf.

9. Adaptive Computing – Documentation / Maui Administrator's Guide. Version 3.2 [Електронний ресурс]. – Режим доступу до ресурсу: http://www.adaptivecomputing.com/resources/docs/maui/index.php.

10. *Управление заданиями в распределенной среде и протокол резервирования ресурсов [Електронний ресурс]. – Режим доступу до ресурсу: http://www.keldysh.ru/papers/2002/prep1/prep2002_1.html#_Toc535316031.*

Надійшла до редколегії 24.03.2012

Рецензент: д-р техн. наук, проф. С.В. Лістровий, Українська державна академія залізничного транспорту, Харків.

ИССЛЕДОВАНИЕ МЕТОДОВ ЛОКАЛЬНЫХ ПЛАНИРОВЩИКОВ РЕСУРСОВ И ИХ МОДИФИКАЦИИ В ГРИД-СИСТЕМАХ

С.В. Минухин, А.В. Мезенцев

Рассмотрена двухуровневая архитектура планирования ресурсов в Грид-системах. Проанализированы локальные системы управления распределенными вычислениями и методы планирования ресурсов. Проанализированы алгоритмы локального планирования ресурсов на примере планировщика Maui. Рассмотрена программная база системы управления распределенными вычислениями. Исследовано расширени возможностей планировщика Maui путем интеграции в программное обеспечение новых алгоритмов планирования ресурсов.

Ключевые слова: *ресурсы, Грид-система, система управления пакетной обработкой, PBS, кластер, Maui, локальный планировщик.*

INVESTIGATION OF METHODS FOR LOCAL RESOURCE SCHEDULING AND THEIR MODIFICATIONS IN GRID SYSTEMS

S.V. Minukhin, O.V. Miezientsev

The two-level architecture of Grid systems resource scheduling is considered. Analyzed the local management of distributed computing and resource scheduling methods. Scheduling algorithms for local resource scheduling example Maui is analyzed. The software base management system distributed computing is consider. The empowerment of Maui scheduler on the base of integration of new software algorithms for resource scheduling is investigated.

Keywords: *resources, Grid system, the control packet processing, PBS, Cluster, Maui, local scheduler.*