УДК 004.75: 004.4'2

О.А. Пріла

ФРЕЙМВОРК ДЛЯ РОЗРОБКИ ВИСОКОРІВНЕВИХ GRID-ЗАСТОСУВАНЬ

Розглянуто особливості використання Grid-середовища для вирішення різних типів обчислювальних завдань. Визначені вимоги до засобів розробки високорівневих Grid-застосувань та наведено результати дослідження існуючих. Запропоновано розширення архітектури відкритого фреймворку gUSE/WS-PGRADE сервісами оптимізації планування та виконання різних типів завдань у Grid-середовищі шляхом аналізу особливостей структури завдання, стану та QoS рівня ресурсів Grid-мережі.

Вступ

Останнім часом Grid-технології активно розвиваються та застосовуються для вирішення обчислювальних завдань великої розмірності. Однак існує проблема використання Grid-середовища фахівцями різних наукових галузей, що обумовлена відсутністю повнофункціональних високорівневих засобів розробки Grid-застосувань та складністю низькорівневих інструментів.

Існує велика кількість прикладних завдань, що характеризуються досить складною структурою та потребують засобів представлення у вигляді потоку робіт (workflow) – обчислювальний сценарій, що складається з окремих обчислювальних кроків, поєднаних у певну послідовність виконання [1]. Планування робочого потоку в загальному випадку не вирішується за допомогою традиційних методів планування поодиноких завдань у Grid-мережі, оскільки має враховувати різнорідність складових завдання, зв'язки між обчислювальними блоками та витрати на пересилання даних між ними.

На рівні проміжного програмного забезпечення Grid (ППЗ) не надається повноцінної підтримки Grid-завдань типу потік робіт. Проблема інтеграції різного ППЗ ускладнює використання всіх доступних користувачу ресурсів для вирішення завдання.

Важливою складовою використання Grid-середовища є надання заданого користувачем рівня якості обслуговування (QoS), що для некомерційного Grid-середовища найчастіше визначається гарантованим часом успішного завершення обчислень, водночас як для комерційної інфраструктури має враховуватись ще й вартість виконання обчислень.

Розробка високорівневих засобів створення Grid-застосувань з наданням механізмів проектування, планування та контролю виконання Grid-завдань різних типів є актуальною задачею сьогодні. Важливою складовою також є наявність інструментів відновлення у випадку збоїв у роботі вузлів мережі та динамічного перерозподілу завдання в разі необхідності.

Метою роботи є формування вимог та дослідження існуючих засобів створення високорівневих Grid-застосувань, а також проектування архітектури фреймворку для розробки Grid-застосувань з підтримкою різних типів обчислювальних задач великої розмірності.

Особливості використання Grid-інфраструктури для вирішення різних типів обчислювальних задач

Одним із факторів, що впливають на продуктивність Grid-мережі, є ефективність планування – рівномірна завантаженість всіх вузлів обчислювальної мережі та мінімальний час простою завдань у черзі на виконання. Ефективність використання Grid-інфраструктури для вирішення завдань різного типу має певні особливості. Відомі наступні типи Grid-завдань: послідовні, паралельні, обробки даних, потоки робіт (workflow).

Ефективність виконання завдання, представленого єдиним обчислювальним блоком або набором послідовних, залежить від ефективності його програмної реалізації, а також стратегії планування брокерів проміжного програмного забезпечення Grid та локального планувальника.

У разі якщо завдання може бути представлено у вигляді набору однотипних завдань з різними вхідними даними, планування зводиться до оптимізації декомпозиції задачі з урахуванням поточних параметрів доступної Grid-інфраструктури та рівня QoS.

Наявність паралельних блоків завдання, що представлено у вигляді потоку робіт, дозволяє одночасно використовувати декілька розподілених ресурсів для більш ефективного вирішення завдання. При цьому мають враховуватись витрати на пересилання даних між ресурсами у відповідності з пропускною здатністю мережі. Витрати на пересилання даних можуть бути усунені шляхом кластеризації декількох блоків потоку робіт для вирішення на одному ресурсі. Існує поняття лінійної та нелінійної кластеризації [2], коли групуються послідовні або паралельні блоки відповідно. Задача оптимізації зводиться до знаходження оптимального рішення між розпаралелюванням та кластеризацією. При цьому має враховуватись зернистість (granularity) завдання. Завдання є крупнозернистим, якщо розмірність обчислень є значно більшою за відношенням до розмірності даних, що пересилаються [2]. Властивість крупнозернистості є необхідною для Grid-завдань.

Завдання типу потік робіт зазвичай представляють у вигляді орієнтованого ациклічного графу (Directed acyclic graph, DAG), вузли якого – блоки обчислень, а дуги – залежності між ними (рис. 1) [2]. Для ефективного планування, окрім стандартних, мають бути визначені наступні параметри блоку:

{ECT, Memory, {T}},

де ECT – передбачуваний час виконання обчислень; Memory – вимоги до пам’яті;

{T} – множина зв’язків з іншими вузлами (односторонній зв’язок).

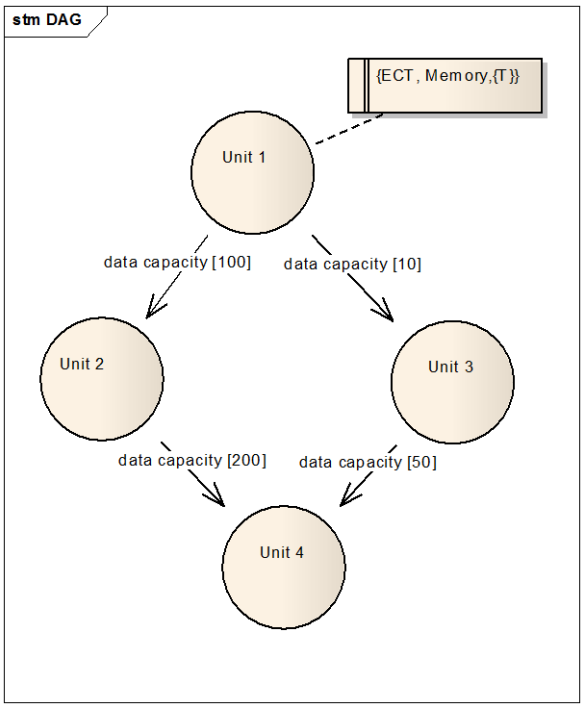

Рис. 1. Приклад структури робочого потоку

Дуги графу характеризуються параметром обсягу даних, що передаються між вузлами. Наявність циклів та розгалужень для моделі потоку робіт не розглядається, оскільки зазначені зв’язки не виправдані з точки зору ефективності використання Grid-середовища.

Структура Grid-мережі може бути представлена у вигляді спрямованого графу, вершини якого – ресурси, а дуги характеризують мережу передачі даних між ними (рис. 2). Кожен вузол структури Grid-мережі характеризується наступними обов’язковими параметрами:

{CPU, Memory, QueueSize, Cost, {R}},

де CPU – обчислювальна потужність; Memory – характеристики пам’яті; QueueSize – розмір черги; Cost – вартість використання; {R} – множина зв’язків з іншими ресурсами (двонаправлений зв’язок).

Задача планування потоків робіт є NP-повною задачею у загальному випадку [2].

На рівні ППЗ не надається повноцінної підтримки Grid-завдань різного типу. Наприклад, ARC Nordugrid (http://www.nordugrid.org/) та gLite (http://glite.cern.ch/), що включені як одні

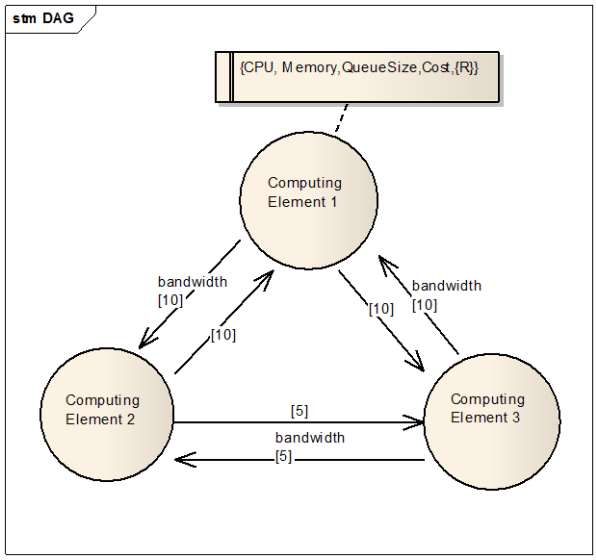

Рис. 2. Приклад структури Grid-мережі

з основних постачальників ППЗ у EMI (http://www.eu-emi.eu/) та найбільш широко використовуються в українській національній Grid-інфраструктурі використовують наступні формати специфікації Grid-завдань: JSDL [3], xRSL [4] та JDL [5]. Серед зазначених, лише JDL-формат вводить поняття типу завдання (Job, DAG та Collection), однак має певні обмеження – не надається можливість визначення періодичної синхронізації між блоками та обсягів даних, що пересилаються. Формати JSDL та xRSL надають лише засоби визначення параметрів поодиноких завдань, життєвий цикл потоку робіт, так само як і зв'язки між окремими завданнями, не підтримуються.

Стратегії планування брокерів ППЗ є дуже спрощеними та не дозволяють виконувати планування потоків робіт з урахуванням особливостей завдання та параметрів QoS. Наприклад, брокер ARC Nordugrid реалізує наступні політики вибору доступних обчислювальних ресурсів: RandomBroker – випадковий вибір; BenchmarkBroker – згідно параметра продуктивності ресурсу; FastestQueueBroker – згідно параметра розміру черги завдань на обчислення; DataBroker – згідно наявності в кеші обчислювального ресурсу даних, необхідних для виконання обчислень [4]. Отже, механізми планування складних Grid-завдань мають бути реалізовані та розташовані поверх рівня ППЗ.

Формування вимог до засобів розробки високорівневих Grid-застосувань. Огляд існуючих

Останнім часом активного розвитку набули високорівневі засоби проектування та розробки Grid-застосувань, що найчастіше являють собою фреймворки для розробки Grid-порталів та desktop-застосувань або системи керування робочими потоками (СКРП, Workflow Management System, WMS). Під СКРП розуміється програмна система, яка відповідає за автоматизоване проектування та виконання потоків робіт, описаних певною вхідною мовою (текстовою або графічною) [1].

Оскільки інфраструктури віртуальних організацій для різних галузей науки є схожими, можливо окреслити базові вимоги до фреймворку для створення високорівневих Grid-застосувань різних типів. Функціональні вимоги наведено далі.

1. Надання API для виконання базових Grid-операцій: створення проксі-сертифікату, перегляд доступних обчислювальних ресурсів та сховищ даних, запуск задачі, моніторинг виконання в динамічному режимі, отримання результатів та ін.

2. Автоматизована генерація файлу специфікації задачі потрібного формату згідно визначених користувачем параметрів задачі та вимог до ресурсів.

3. Підтримка різних типів завдань з наданням графічних компонентів для визначення обчислювальних блоків та схеми синхронізації між ними.

4. Підтримка періодичної синхронізації між паралельними блоками обчислень, коли один блок може розпочинати обчислення як тільки підготовлено частковий набір даних іншого блоку.

5. Можливість визначення вимог до рівня обслуговування. Для забезпечення потрібного рівня QoS необхідно отримання угоди про рівень якості від про-

вайдерів послуг. У випадку з некомерційними Grid-інфраструктурами така інформація зазвичай не надається. Тому необхідний модуль, який би дозволив отримувати статистику вже виконаних завдань і аналізувати рівень QoS.

6. При плануванні завдань має враховуватись встановлений рівень QoS-вимог та поточний стан Grid-мережі, а також можливість динамічного перерозподілу блоків завдання під час виконання в зв'язку зі змінами параметрів Grid-мережі.

7. Наявність механізму контрольних точок.

8. Надання можливості попереднього замовлення (advanced reservation) Grid-ресурсів.

9. Можливість збереження образу Grid-задачі для повторного запуску без проектування.

10. Визначення потрібного середовища виконання (Runtime Environment, RE) складових задачі. Мають бути реалізовані інструменти вибору образу віртуальної машини для визначення необхідного RE.

11. Надання API для підтримки роботи віртуальних організацій: управління користувачами віртуальної організації; визначення власником тарифікації оплати за користування ресурсом; моніторинг стану ресурсів; облік використання Grid-ресурсів.

Окремо слід виділити нефункціональні вимоги до фреймворку.

1. Підтримка стандартів EMI. З метою інтеграції ресурсів, що працюють під різним ППЗ, потрібен єдиний інтерфейс для виконання запитів. EMI є стандартом де-факто для постачальників Grid. На даний час NorduGrid ARC, gLite, UNICORE та dCache орієнтовані на підтримку стандартів EMI.

2. Кросплатформеність. Можливість розробки на базі фреймворку Grid-застосувань для різних платформ, мобільних пристроїв у тому числі.

3. Розширюваність. Гнучка архітектура для доповнення фреймворку новими модулями, в тому числі підтримки нового middleware, або внесення змін до інтерфейсу існуючого.

4. Маштабованість. Можливість розміщення компонентів Grid-застосувань на розподілених ресурсах з метою підвищення продуктивності.

5. Забезпечення безпеки Grid-порталу та ресурсів ВО.

Були розглянуті системи для проектування потоків робіт (Taverna [6], Montage [7], Triana [8], Kepler [9], WS-VLAM [10], MathCloud [11], MaWo [12], Askalon [13]) та інструменти розробки Grid-застосувань (GridNNN [14], gUSE/WS-PGRADE [15], Lunarc app [16], Ganga [17], GridSphere Portal Framework [18], SimpleGrid [19], VGrADS [20], Pegasus [21], gEclipse [22]). Деякі з розглянутих засобів являють собою системи управління бізнес-процесами та реалізують розвинутий функціонал для проектування потоків робіт, однак не надають функціоналу з виконання базових Grid-операцій, і хоча передбачають механізми інтеграції з розподіленим середовищем, вони досить складні в реалізації. Деякі з систем спеціалізовані для вирішення завдань певної наукової галузі, наприклад, MathCloud, WS-VLAM та ін. Порівняльний аналіз систем наведено в працях [1, 23]. Основними проблемами використання існуючих засобів розробки високорівневих Grid-застосувань є: орієнтація на певний тип ППЗ без надання гнучких механізмів розширення; недостатня реалізація функціоналу для виконання базових Grid-операцій; орієнтація на певний клас завдань; відсутність модуля оптимізації виконання Grid-завдань різних типів.

Клієнтська частина та редактор потоків робіт більшості систем представлені віконним інтерфейсом, водночас як більш перспективним є реалізація у вигляді Web-інтерфейсу [1] з точки зору зменшення вимог до клієнтської сторони та відповідно більшої переносимості застосування (portability).

Серед розглянутих інструментів як найбільш функціональні і підтримувані на сьогодні можна зазначити фреймворк

gUSE/WS-PGRADE та СКРП Taverna. WS-PGRADE пропонує гнучкий механізм розширення системи плагінами взаємодії з різним ППЗ. Важливою особливістю є можливість легкої адаптації та налаштування у відповідності з потребами різних Grid-співтовариств у цілях вирішення наукових завдань. Фреймворк надає API для реалізації більшості базових Grid-операцій, однак не реалізує засоби оптимізації виконання потоків робіт з урахуванням встановленого рівня QoS. Розмір задачі та обсяги даних, що пересилаються, не враховуються при плануванні та не можуть бути визначені як параметри задачі. Інструменти проектування потоків робіт не підтримують зв'язки періодичної синхронізації обчислень між паралельними блоками обчислень.

WS-PGrade надає можливість інтеграції з системами керування робочими потоками ASKALON та MOTEUR [24]. Однак зазначені СКРП орієнтовані на використання певного ППЗ, а саме ASKALON орієнтована на використання ППЗ Clobus, система MOTEUR – на використання gLite. Відповідне рішення не є гнучким та універсальним, оскільки не дозволяє залучати для виконання обчислень робочого потоку ресурси, що використовують різне ППЗ.

У роботі пропонується розширення архітектури фреймворку WS-PGRADE механізмами оптимізації планування різних типів завдань у Grid-середовищі.

Проектування фреймворку для створення Grid-застосувань на базі gUSE/WS-PGRADE

Архітектура фреймворку gUSE/WS-PGRADE була розширена сервісами Extended gUSE Services. Загальна архітектура фреймворку представлена на рис. 3.

Фреймворк gUSE/WS-PGRADE надає інтерфейси (Application Specific Module API та Remote API) для розробки високорівневих Grid-застосувань, спеціалізованих для вирішення наукових завдань різних Grid-співтовариств.

Архітектура фреймворку gUSE/WS-PGRADE є тришаровою – шар відображення, шар сервісів gUSE та шар інтеграції з ППЗ розподілених обчислювальних інфраструктур (Distributed Computing Infrastructures, DCIs).

Шлюз DCI надає інтерфейс для розширення фреймворку плагінами підтримки ППЗ та призначений для взаємодії з сервісами проміжного програмного забезпечення Grid, кластерних та хмарних середовищ. Шлюз надає стандартний BES-інтерфейс для відправки завдань.

Web-портал WS-PGRADE реалізує інтерфейс користувача сервісів gUSE.

GUSE (Grid User Support Environment) – це набір високорівневих сервісів та реалізований як набір Web-служб. Компоненти gUSE реєструються в централізованій інформаційній системі (Information System) та можуть бути розподілені на декілька серверів з метою підвищення продуктивності роботи застосування, що доводить масштабованість архітектури фреймворку.

Модуль аутентифікації реалізує сервіси взаємодії та використання функціоналу MyProxy-серверу. Інструменти пошуку віддалених сховищ даних, відправлення та отримання даних з/на ресурс реалізовано в межах компоненту File Storage. gUSE надає API для роботи з репозиторієм образів робочих потоків – визначених параметрів та налаштувань прикладної задачі, що дозволяє співпрацювати користувачам віртуальних організацій над розробкою та адаптацією наукових завдань для вирішення у Grid-середовищі.

Клієнтську частину редактора потоків робіт реалізовано у вигляді desktop-застосування на базі технологій JAVA AWT та Swing (рис. 4). Реалізовано механізми автоматичного формування JDL-специфікації.

Алгоритм обчислювального блоку завдання може бути представлений у вигляді: а) бінарного виконуваного файлу; б) виклику Web-сервісу (з використанням REST-підходу); в) виклику вкладеного потоку робіт.

У межах розроблюваного в роботі фреймворку з розширеним набором серві-

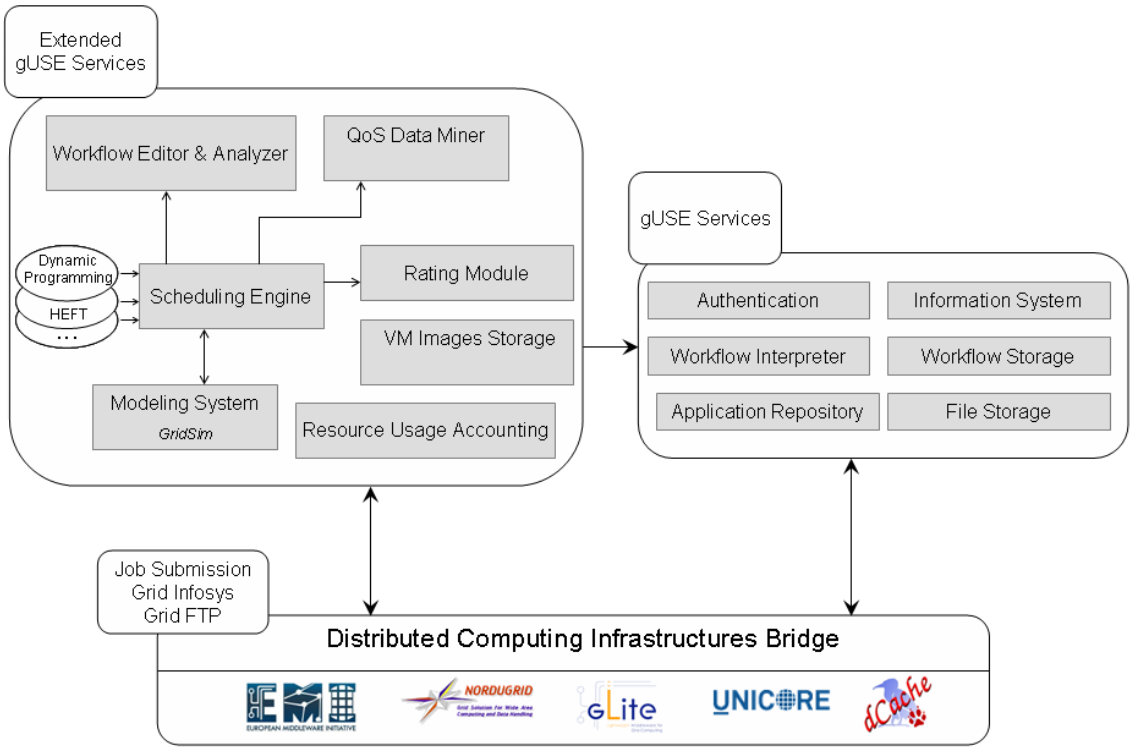

Рис. 3. Архітектура фреймворку для створення Grid-застосувань на базі gUSE/WS-PGRADE

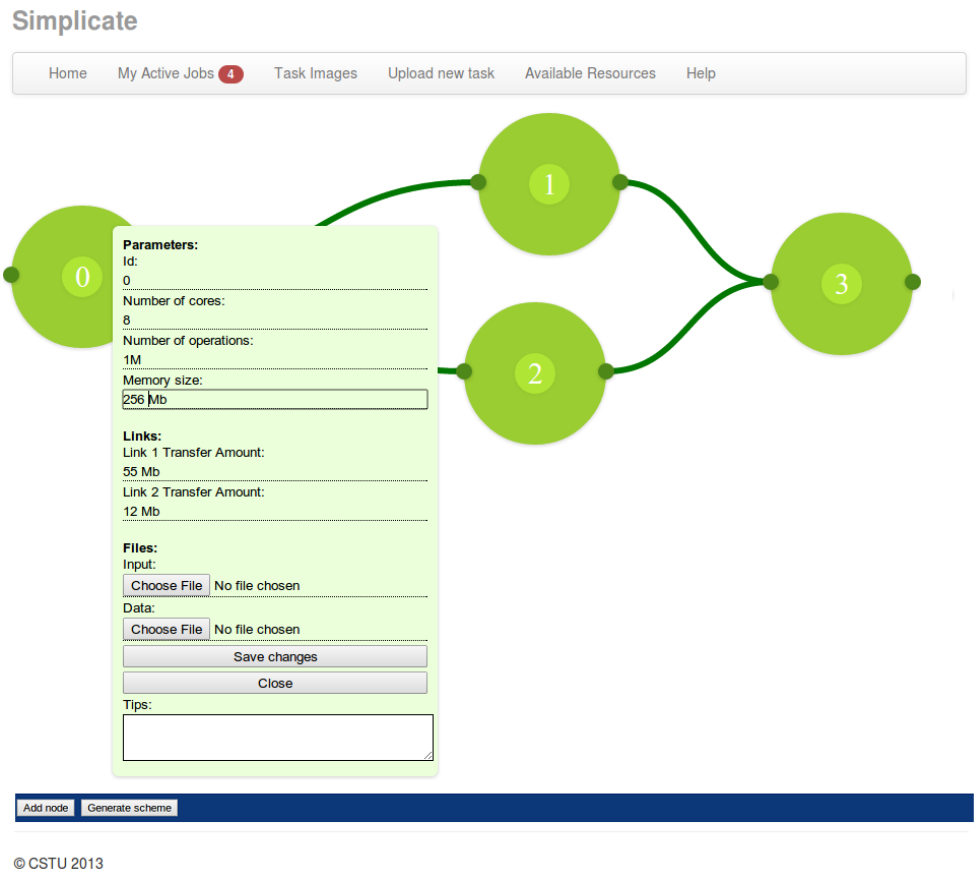

Рис. 4. Редактор потоків робіт

сів реалізовано модуль проектування та аналізу потоків робіт (Workflow Editor and Analyzer). Клієнтську частину редактора потоків робіт реалізовано з використанням JavaScript бібліотек D3.js [25] та helios.js [26] з метою зменшення вимог до програмного забезпечення клієнтської машини (рис. 4). Окрім стандартних, надається можливість визначення таких параметрів завдання як приблизна оцінка розмірності обчислень, обсяги вхідних/вихідних даних та QoS параметри.

Введено поняття шаблону робочого потоку та реалізовані наступні шаблони: паралельна обробка даних; однорідний/неоднорідний робочий потік; збалансовані/незбалансовані робочі потоки. Реалізовані механізми верифікації та автоматизованої типізації спроектованого потоку робіт.

Модуль планування виконання потоків робіт (Scheduling Engine) реалізує механізм оптимізації вибору ресурсів з урахуванням шаблону та параметрів Grid-завдання, а також даних про рівень якості послуг ресурсів QoS. На даний момент реалізовано евристичний алгоритм планування на основі списків з урахуванням залежностей Heterogeneous-Earliest-Finish-Time (HEFT) [2] та запропонований авторами алгоритм планування на базі методу динамічного програмування, що враховують час виконання обчислювального блоку на певному ресурсі, час очікування у черзі до ресурсу, час на пересилання даних між обчислювальними блоками та вартість використання Grid-ресурсу. Вигляд цільової функції залежить від запитуваного рівня QoS – пріоритет часу або вартості. В загальному випадку цільова функція є багатокритеріальною:

$$\varphi = f\ \left(time,\ cost\right) \rightarrow min. \qquad (1)$$

Обидва алгоритми характеризуються лінійною складністю. З метою оцінки ефективності алгоритмів планування, а також отримання приблизної оцінки часу виконання завдання в режимі реального часу було реалізовано імітаційну модель виконання завдань різного типу у Grid-середовищі на базі інструменту GridSim [27] (Modeling System), архітектуру якого було розширено модулями представлення моделі завдання у вигляді потоку робіт та представлення моделі гетерогенної Grid-мережі.

Використовується динамічний підхід до планування потоків робіт, що враховує динаміку виконання завдання (стан виконання складових завдання, врахування ситуацій, коли завдання може бути витіснено іншим, що має більш високий пріоритет) та динамічність Grid-мережі (стан та завантаженість ресурсів, змінення локальних політик розподілення ресурсів). Модуль планування надає можливість попереднього замовлення ресурсів для тих Grid-сайтів, що підтримують зазначену політику виділення ресурсів.

З метою оцінки якості послуг Grid-ресурсів або коригування рівнів QoS, отриманих на підставі деякого SLA (Service-level agreement), розробляється модуль збору та аналізу рівня QoS ресурсів Grid-мережі (QoS Data Miner). Розглядаються такі типи даних про рівень послуг – узагальнені дані (пропускна спроможність мережі, обчислювальна потужність вузлів, обсяг дискового простору тощо) та неявні дані про рівень якості послуг (завантаженість та частота відмови вузлів, час простою завдань в черзі, змагання за певний ресурс на вузлах між незалежними задачами). Більшість узагальнених даних можливо отримати з використанням інформаційних сервісів ППЗ (GLUE2 XML та LDAP). Для визначення неявних параметрів використовується механізм періодичного запуску «порожніх» програм-обгорток, що не виконують обчислень, однак дозволяють отримати інформацію про стан Grid-мережі. Для аналізу рівня QoS також використовується статистика виконання реальних завдань в Grid. Для кластеризації та подальшого аналізу статистичної інформації, представленої у вигляді лог-файлів, використовується бібліотека AfterGlow [28]. Модуль автоматичного визначення пріоритетів Rating Module на основі статистики виконання завдань на певному ресурсі встановлює коефіцієнти значимості показників QoS.

Можливість визначення як середовища виконання завдання образу віртуальної машини, що має бути розгорнута на ресурсі для виконання обчислень, реалізується сервісами модуля зберігання образів VM Images Storage на базі системи oVirt [29]. Сервіси обліку використання обчислювальних ресурсів та здійснення нарахувань згідно встановлених власником тарифів реалізовані в межах модуля Resource Usage Accounting на базі інтерфейсів ППЗ APEL та DGAS client.

Висновки

В роботі розглянуті особливості використання Grid-інфраструктури для вирішення різних типів обчислювальних завдань, на базі чого сформовані вимоги до засобів розробки високорівневих Grid-застосувань. Наведені результати дослідження існуючих інструментів розробки Grid-застосувань та запропоновано розширення архітектури фреймворку gUSE/WS-PGRADE модулями планування виконання завдань різного типу з урахуванням QoS та поточного стану Grid-мережі. Сервіси проектування робочих потоків розширені можливістю визначення розмірності обчислень та даних, що пересилаються, а також визначення образу віртуальної машини як середовища виконання обчислювального блоку завдання. Передбачаються механізми верифікації та типізації структури робочих потоків.

Фреймворк розробляється в межах відкритого проекту. Результат буде представлений у вигляді java-бібліотек та прикладу застосування фреймворку для створення Grid-порталу віртуальної організації економічного прогнозування. Застосування фреймворку дозволить зменшити витрати при розробці кросплатформенних Grid-застосувань та підвищити ефективність використання Grid-середовища для вирішення прикладних задач різних наукових галузей.

1. *Петренко А.І., Булах Б.В.* Застосування workflow-систем для потреб сучасних науки та інженерії // Наукові вісті НТУУ "КПІ". – 2011. – № 5. – С. 40–51.
2. *Alberto Forti.* DAG Scheduling for grid computing systems. Ph.D. Thesis, University of Udine – Italy, Department of Mathematics and Computer Science, 2005–2006.
3. *Job* Submission Description Language (JSDL) Specification, Version 1.0, 2008, GFD-R.136.
4. *Extended* Resource Specification Language, Reference Manual for ARC versions 0.8 and above, Nordugrid-Manual-4, 2013.
5. *Job* description language attributes specification for the gLite Workload Management System, 2011, WMS-JDL.doc.
6. *Hull D., Wolstencroft K., Stevens R., Goble C., Pocock M.R., Li P., and Oinn T.* Taverna: A tool for building and running workflows of services. In: Nucleic Acids Research. – 2006. – Vol. 34. – P. W729–W732,
7. *Joseph C. Jacob, Daniel S. Katz, G. Bruce Berriman, John Good, Anastasia C. Laity, Ewa Deelman, Carl Kesselman, Gurmeet Singh, Mei-Hui Su, Thomas A. Prince, Roy Williams.* Montage: a grid portal and software toolkit for science-grade astronomical image mosaicking. In: Int. J. Computational Science and Engineering. – 2009. – P. 73–87.
8. *Taylor I., Shields M., Wang I., and Harrison A.* Visual Grid Workflow in Triana. In: Journal of Grid Computing. – 2005. – Vol. 3, N 3-4. – P. 153–169.
9. *Ludäscher B., Altintas I., Berkley C., Higgins D., Jaeger E., Jones M., Lee E.A., Tao J., and Zhao Y.* "Scientific workflow management and the kepler system: Research articles," Concurr. Comput.: Pract. Exper., – 2006. – Vol. 18, N 10. – P. 1039–1065.
10. *Korkhov V.V., Vasyunin D.A., Belloum A.S.Z., Andrianov S.N., Bogdanov A.V.* Virtual Laboratory and scientific workflow management on the grid for nuclear physics applications. In: Distributed computing and grid-technologies in science and education, Proceedings of the 4th lnternational Conference, Dubna, 2010. – P. 153–158.
11. *Voloshinov V.V., Smirnov S.A.* Error-free inversion of ill-conditioned matrices in distributed computing system of restful-services of computer algebra. In: Distributed computing and grid-technologies in science and education, Proceedings of the 4th lnternational Conference, Dubna, 2010. – P. 257–263.
12. *Sukhoroslov O.V.* On development of grid-enabled applications and service-oriented scientific environments. In: Distributed computing and grid-technologies in science and education, Proceedings of the 4th lnternational Conference, Dubna, 2010. – P. 236–240.
13. *Marek Wieczorek, Radu Prodan and Thomas*

Fahringer. Scheduling of Scientific Workflows in the ASKALON Grid Environment. In: ACM SIGMOD Record Journal, Sept. – 2005. – Vol. 34, N 3. – P. 56–62.
14. *Shamardin L., Demichev A., Kryukov A., Ilyin V.* GRIDNNN job execution service: a restful grid service. In: Distributed computing and grid-technologies in science and education, Proceedings of the 4th lnternational Conference, Dubna, 2010. – P. 215–219.
15. *Kacsuk P.*: P-GRADE portal family for Grid infrastructures, Concurrency and Computation: Practice and Experience Journal. – 2011. – Vol. 23, N 3. – P. 235–245.
16. *Appleton O., Cameron D., Cernak J., Dóbé P., Ellert M., Frågåt T., Grønager M., Johansson D., Jönemo J. and others.* The next-generation ARC middleware. Ann. Telecommun. – 2010. – 65:771–776, DOI 10.1007/s12243-010-0210-2.
17. *Ganga.* Simplifying use of the Grid [Електронний ресурс]. – Режим доступу: http://ganga.web.cern.ch/ganga.
18. *Install* GridSphere Portal Framework [Електронний ресурс]. – Режим доступу: http://technical.bestgrid.org/index.php/Install_GridSphere_Portal_Framework.
19. *Projects:*simplegrid:index [CyberInfrastructure and Geospatial Information Laboratory] [Електронний ресурс]. – Режим доступу: https://www.cigi.uiuc.edu/doku.php/projects/simplegrid/index.
20. *Lavanya Ramakrishnan, Charles Koelbel et al.* VGrADS: Enabling e-Science Workflows on Grids and Clouds with Fault Tolerance [Електронний ресурс]. – Режим доступу: http://citeseerx.ist.psu.edu/viewdoc/similar?doi=10.1.1.160.7546&type=sc/.
21. *Ewa Deelman, James Blythe et al.* Pegasus: Mapping scientific workflows onto the grid, 2004 [Електронний ресурс]. – Режим доступу: https://www.cct.lsu.edu/~kosar/csc7700-fall06/papers/Deelman04.pdf.
22. *g-Eclipse* Project – Tools for Grid and Cloud Computing [Електронний ресурс]. – Режим доступу: http://www.eclipse.org/geclipse/.
23. *Curcin V., Ghanem M.* Scientific workflow systems – can one size fit all? Proceedings of the 2008 IEEE, CIBEC'08.
24. *Tristan Glatard, Johan Montagnat, Diane Lingrand, Xavier Pennec.* Flexible and efficient workflow deployement of data-intensive applications on grids with MOTEUR // In: International Journal of High Performance Computing Applications (IJHPCA), 22 (3), SAGE, August 2008. – P. 347–360.
25. *D3.js* – Data-Driven Documents [Електронний ресурс]. – Режим доступу: http://d3js.-org/.
26. *HeliosJS* by entrendipity [Електронний ресурс]. – Режим доступу: http://entrendipity.github.io/helios.js/.
27. *Rajkumar Buyya and Manzur Murshed.* GridSim: A Toolkit for the Modeling and Simulation of Distributed Resource Management and Scheduling for Grid Computing, The Journal of Concurrency and Computation: Wiley Press, Nov.-Dec., 2002. – Practice and Experience (CCPE), Vol. 14, N 13-15,
28. *AfterGlow* – Link Graph Visualization [Електронний ресурс]. – Режим доступу: http://afterglow.sourceforge.net/.
29. *Офіційний* сайт oVirt [Електронний ресурс]. – Режим доступу: http://www.ovirt.org.

Одержано 11.07.2013

Про автора:

Пріла Ольга Анатоліївна,
асистент кафедри.

Місце роботи автора:

Чернігівський національний
технологічний університет,
м. Чернігів, вул. Шевченка 95.
Тел..: (093) 787 1972.
E-mail: olga.prila1986@gmail.com