Библиотечные компоненты можно многократно применять в проектах. Библиотека стандартных деталей (компонентов), поставляемая вместе с autodesk inventor, содержит более 700 тыс. Элементов моделей различных стандартов. Это – гайки, болты, винты и другие детали. Сюда включены стандартные детали трубопроводов, отводов, стальных профилей, подшипников, деталей крепления.

Существует возможность создавать в библиотеке компонентов новые разделы и публиковать в них собственные модели, предварительно сформированные в виде параметрических рядов.

Публикация в библиотеку компонентов стандартных деталей, используемых на предприятии, позволяет гарантировать, что все пользователи будут работать с одними и теми же стандартными деталями и узлами.

Учитывая вышеизложенное, целями работы являются:

- Создание и редактирование параметрического ряда для нестандартных деталей;
- Подготовка моделей, сформированных в виде параметрических рядов, к публикации в библиотеку компонентов;
- Публикация моделей в библиотеку компонентов.

В работе рассмотрена технология создания параметрического ряда детали. Перед построением параметрического ряда необходимо создать пользовательскую библиотеку компонентов на сервере vault для возможности ее коллективного использования. Затем в проекте inventor выполнить подключение созданной библиотеки для чтения и записи.

Создание параметрического ряда деталей заключается в определении нескольких экземпляров детали с заданием параметров и свойств каждого из них. Для удобства использования данные организованы в виде таблицы. Совокупность деталей, генерируемых таблицей, образует параметрический ряд. С помощью внешнего приложения ms excel могут быть добавлены остальные типоразмеры параметрического ряда в соответствии со стандартом.

Подготовка параметрического ряда для публикации в библиотеку компонентов заключается в том, что параметрический ряд детали (модели) переносится в специальный механизм (сапр) «создание компонента»

Для публикации готовой параметрической модели детали в библиотеку компонентов необходимо выбрать на ленте сапр «управление» команду «публикация детали». После завершения процесса публикации новое семейство деталей доступно для использования в библиотеке компонентов.

В работе показаны конкретные примеры разработки отдельных нестандартных деталей и помещения их в библиотеку компонентов.

Лаврик С. Е. (ХАІ), Коваленко М. А.,
Кондратюк В. А., Горбач А. В.,
Кошлатий О.В. (УкрДАЗТ)

СТРАТЕГІЇ РОЗПОДІЛУ РЕСУРСІВ І ПЛАНУВАННЯ В ГЕТЕРОГЕННИХ СЕРЕДОВИЩАХ ГРІД СИСТЕМ З ВИКОРИСТАННЯМ ЗАВДАНЬ ПРО МІНІМАЛЬНЕ ПОКРИТТЯ НЕЛІНІЙНОГО БУЛЕВОГО ПРОГРАМУВАННЯ І ПРОЦЕДУР ТИПУ FCFC

Введення

Безпеку на залізниці не можна розглядати окремо від конкретної служби чи господарства, а оскільки існує явна стратифікація, то необхідно визначити якийсь спосіб системної інтеграції, який може полягати у виділенні бізнес-правил, що пов'язують ці шари або служби, при цьому, природно, враховують аспект безпеки . Інфраструктура залізничного транспорту розглядається як організм, що володіє певним рівнем життєздатності. У цьому організмі, як і в будь-якому живому, відбуваються різномасштабні процеси на різних рівнях. У зв'язку з цим найбільш перспективним видається виділення багатоконтурних систем управління за принципом однорідності інтегровних підсистем щодо масштабу часу. Даний принцип, зокрема, передбачає виділення, принаймні, двох контурів, хоча їх, безумовно, значно більше. Перший контур - контур оперативного рівня або контур реального часу, в якому відбувається циркуляція інформації за малі проміжки часу. На цьому рівні масштаб часу між подіями такий, що час прийняття рішень дуже малий і рішення покладаються на автомат (системи автоматики локомотива, СЦБ та ін.) Другий контур - псевдореального часу або часу прийняття управлінських рішень, що вимірюється в годинах, добі, де в контур зворотного зв'язку може бути включена людина. На цьому рівні приймаються оперативні рішення, які вимагають рівня інтелекту людини. Для ефективного управління на цьому рівні необхідна наявність системи підтримки прийняття рішень. Ця підтримка заснована на експертній інформації, концентрованої в системі управління знаннями. Одна з найбільш актуальних проблем - це наявність великого числа не пов'язаних один з одним систем, які вирішують якусь вузьку задачу. Це обумовлено історичними факторами. У процесі розвитку системи управління залізничним транспортом при вирішенні окремо взятих завдань із забезпечення безпеки на тій чи іншій ділянці управління перевезеннями виникала потреба використовувати ті чи інші апаратно програмні засоби автоматизації. У результаті накопичилася велика кількість локальних рішень, виникла так звана «клаптева автоматизація».

Так, наприклад, на тяговому рухомому складі існує ряд пристроїв, що реалізують функції забезпечення безпеки руху: система автокерування поїзда, автоматичного гальмування, комплексний локомотивний пристрій безпеки, пристрій контролю пильності машиніста. Існують численні системи, пристрої, прилади залізничної автоматики і телемеханіки, автоматизовані системи управління перевезеннями різних рівнів, контролю дислокації рухомого складу і локомотивних бригад і т.д. Також існує велика кількість автоматизованих робочих місць і систем управління різних господарств, інформаційних систем, мереж, серверів і баз даних. Істотний резерв підвищення безпеки на залізничному транспорті - об'єднання окремих підсистем в єдину багаторівневу систему. Безпека перевезень при цьому повинна розглядатися, насамперед, як безпека внутрішнім управлінням тяговим рухомим складом, керування поїзда в цілому, управління рухом поїздів на полігоні у взаємодії з іншими підсистемами. Сучасні технічні засоби та інформаційні технології дозволяють реалізувати це завдання з мінімальними витратами. Однак необхідно реалізувати складову, що забезпечує інформаційну ув'язку систем один з одним, їх узгоджене функціонування і керування, що реалізує взаємодію з інформаційними системами залізничного транспорту. Іншими словами, необхідно створити системоутворюючу середу, в яку включені вже створені системи. Ця середа повинна забезпечити від взаємодії окремих підсистем нову якість, спрямовану на підвищення безпеки руху поїздів. Крім того, система управління безпекою є територіально-розподіленою системою, що обумовлює необхідність створення специфічної мережевої архітектури заснованої на грід технологіях. Існуючий стан системи визначається в результаті активного моніторингу, який є результатом логічного висновку на правилах, що складають модель безпеки. Побудова такої системи вимагає розробки математичних моделей технічного стану рухомого складу і об'ємного планування їх технічного обслуговування як основних складових концепції безпеки. Оцінка технічного стану рухомого складу повинна будуватися на тому припущенні, що процес експлуатації з урахуванням всіх діючих факторів носить випадковий характер, і, як наслідок, дослідження технічного стану має базуватися на теорії математичної статистики. Складні системи являють собою багатоконтурні системи, що реалізують як підлегле регулювання одного процесу, так і паралельне управління різними функціями системи. Стосовно до задачі управління безпекою на залізничному транспорті можна говорити, що існує складнопідрядна система управління кожним із господарств: локомотивним, вагонним, колійним, СЦБ та ін. Багаторівневу систему управління і забезпечення безпеки руху поїздів можна визначити як сукупність технічних і організаційних заходів щодо управління та здійснення перевезень, що дозволяють забезпечити безпеку руху поїздів як в кожному елементі кожного контуру управління, так і в системі управління в цілому. Необхідність дублювання, резервування та ін. в кожному окремому випадку визначається індивідуально, виходячи з аналізу небезпеки наслідків відмови елемента, контуру та системи в цілому. Аналіз технічної ефективності впровадження автоматизованих систем управління в господарствах залізниці, оцінка витрат необхідних для розробки нормативних документів, технологій роботи та програмного забезпечення аналітичних систем різних рівнів, впровадження системи, навчання персоналу дозволяють стверджувати, що пропоновані рішення, спрямовані на забезпечення взаємодії без зміни алгоритмів роботи суміжних систем, порівняно з іншими можливими є в два рази дешевше і в три рази швидше за термінами реалізації. Таким чином, найбільш доцільним інформаційним середовищем яке задовольнить всі перераховані вимогами до створення автоматизованих систем безпечного управління залізничним транспортом можуть з'явитися інформаційні системи на основі грід технологій.

Грід технології. На сьогодні найбільші успіхи досягнуті в області грідо обчислювального типу, тому в якості платформи для обговорення перспектив обрано, мабуть, саме розвинене проміжне програмне забезпечення з цієї області - комплекс gLite (www.glite.org). Як відомо, це проміжне програмне забезпечення стає основним для інфраструктури проекту EGEE (www.eu-egee.org), приходячи на зміну пакету LCG [1]. Характеризуючи gLite, слід враховувати, що його розробка не завершена, тому ми не будемо зосереджуватися на недоліках конкретного релізу, а розглянемо цей комплекс з точки зору його функціональних можливостей.

Можливості gLite визначаються трьома групами технологій:

- інтеграції та віртуалізації ресурсів;
- підтримки функціонування гріду;
- побудови розподілених додатків.

Розглянемо коротко суть цих технологій. Можливість будувати комплексні програми, в яких поєднується оперативна поставка даних, зберігання великих масивів і автоматична обробка високого рівня складності.

Інтеграція комп'ютерних ресурсів і ресурсів зберігання насамперед служить засобом агрегування великих обсягів ресурсів для їх колективного і, як наслідок, економного використання. Це наділяє програми, засновані на цих технологіях, здатністю забезпечувати масове і оперативне обслуговування великого числа користувачів. Ці ж технології можуть бути засобом забезпечення надійності додатків, оскільки за допомогою механізму віртуалізації може

здійснюватися реплікація даних і використання альтернативних ресурсів. Інтеграція джерел даних, в якості яких можуть виступати датчики.

Програмні засоби моніторингу, а також призначені для користувача інтерфейси:

- дозволяють збирати інформацію з безлічі різних місць;
- робить її потенційно доступної повсюдно;
- дозволяє здійснювати адресну доставку інформації, з урахуванням ролей і праводержувачів.

Технології підтримки функціонування гріду. Перейдемо до другої групи технологій - підтримки функціонування гріду. У попередників комплексу gLite - DataGrid [4], Condor [5], LCG вперше стала проводитися систематична лінія на те, що мало реалізувати засоби виконання якихось змістовних операцій (запитів на управління завданням, або передачу файлів). У розподіленому середовищі сам процес виконання операцій повинен підтримуватися спеціальним чином. Це дало поштовх до розвитку групи підтримуючих технологій, яка включає наступне. Протоколювання: обробка завдань проводиться різними компонентами програмного забезпечення, які в розподіленому середовищі знаходяться в різних місцях. *Протоколювання.* Здійснює збір діагностичної інформації і робить її доступною в цілому. Застосовується механізм, який: 1) відстежує завдання в кожній точці, збираючи діагностичні повідомлення обробних програм і записуючи їх в локальну базу даних (БД), 2) періодично передає зібрану локальну інформацію в загальну БД, де вона агрегується. Моніторинг завдань, дозволяє отримати в кожен момент статус обробки завдання. Цінність цієї функції не тільки в інформуванні користувача, а й у забезпеченні надійності: при помилці на якомусь етапі, повинен бути проінформований попередній, і, якщо збій викликаний технічними причинами, обробка може бути відновлена на альтернативних компонентах інфраструктури. Оскільки взаємодія компонент не припускає, що викликаюча очікує закінчення обробки і отримує код повернення, механізм моніторингу заснований на періодичному опитуванні стану з викликаючого рівня на викликаний.

Моніторинг ресурсів. Збирає статичну (характеристики) і динамічну (стан) інформацію про ресурси. Виходячи з цієї інформації, здійснюється віртуалізація ресурсів: визначається можливість їх використання, справність і завантаженість. Реалізується у вигляді програм-сенсорів (джерела даних), розташованих в середовищі ресурсів і поставляючих дані в БД.

Облік споживання ресурсів. Детально реєструє кількість ресурсів, спожитих завданнями при їх виконанні. Збір цих даних необхідний для вибудовування економічних відносин між постачальниками ресурсів та споживачами. Як і моніторинг, поставка даних здійснюється сенсорами.

Перспективи застосування гріду та технології побудови розподілених додатків. Розглянемо тепер gLite з точки зору його застосування. gLite (і подібні до нього системи) - це комплекс, призначений для управління прикладними програмами в розподіленому середовищі ресурсів. Діяльність ця підтримується багатопланово, але сама вона є специфічною - це комп'ютинг, орієнтований на науково-технічну сферу: користувач працює з базовими об'єктами - програмами і файлами, тобто від нього вимагається кваліфікація хоча й прикладного, але все-таки програміста. Комп'ютинг такого роду, безсумнівно, має цінність для вирішення різних і досить важливих прикладних задач, але є всього лише одним з можливих різновидів розподіленого комп'ютингу, які можуть підтримуватися грід-інфраструктурою. Найбільш складною і невирішеною до кінця проблемою є задача планування розподілу ресурсів в гетерогенних грід системах. Завдання планування розподілу ресурсів в гетерогенних середовищах широко досліджується в даний час і є досить великий список робіт у цьому напрямку, проте як видно з експериментальних досліджень проведених в роботі [6] істотним недоліком відомих стратегій розподілу завдань в багато процесорних системах в кінцевому рахунку є нерівномірність їх завантаження , що призводить до зменшення часу виконання всіх завдань у черзі. Відомо, що частина ресурсів системи може бути завантажена на 100-80%, а частина ресурсів не більше ніж на 20%, тобто нерівномірність навантаження дуже велика. Для усунення цього недоліку в даній роботі пропонується наступний підхід.

Черга завдань знаходиться в буфері сервера кластера, при цьому кожне завдання характеризується вектором показників $(\alpha_1^{3_i}, \alpha_2^{3_i}, ..., \alpha_q^{3_i})$, в якості яких може бути об'єм пам'яті необхідний для виконання завдання, вимоги до операційної системи ресурсу, договірною ціною рішення і т.д. Кожен $\{R_i\}$ надає дані відомості на сервер про про те вільний він чи ні і свої характеристики в тому ж форматі, що і для завдань. $(\alpha_1^{R_i}, \alpha_2^{R_i}, ..., \alpha_q^{R_i})$. З буфера сервера формується пул завдань, який потрібно відправити на виконання на ресурси $\{R_i\}$. На основі відомостей про завдання і вільні ресурси кластера, в базі даних сервера створюється таблиця, стовпці якої відповідають ресурсу $\{R_i\}$ кластера, рядкам завдання з сформованого пула завдань. На перетині рядка і стовпця будемо ставити одиницю, якщо з порівняння $(\alpha_1^{3_i}, \alpha_2^{3_i}, ..., \alpha_q^{3_i})$ и $(\alpha_1^{R_i}, \alpha_2^{R_i}, ..., \alpha_q^{R_i})$ і випливає, що дана задача може бути реалізована на наявному вільному ресурсі і нуль в іншому випадку.

Завдання $З_i$ може бути виконане на ресурсі R_i, якщо виконуються нерівності $\alpha_1^{R_i} \le \alpha_1^{З_i}; \alpha_2^{R_i} \le \alpha_2^{З_i}; ..., \alpha_q^{R_i} \le \alpha_q^{З_i}$. Розподіл завдань розглянемо на основі принципу роздільного розподілу завдань див. рис.1, а динамічно і безперервно на основі наступної процедури ***D*** .

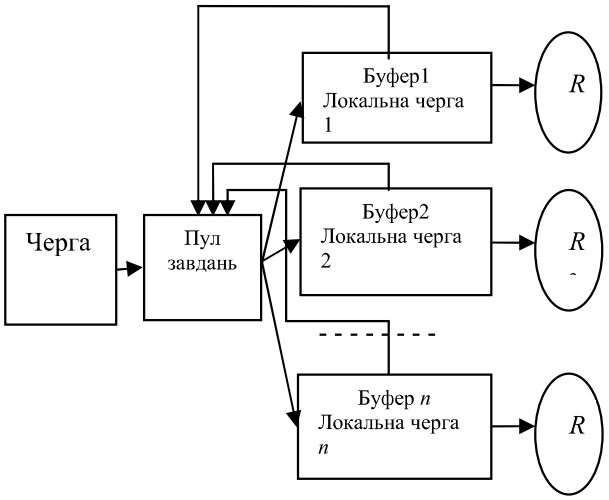

Рисунок 1 – Роздільне розподіл завдань в гетерогенному середовищі

Процедура D

На основі відомостей про завдання і вільних ресурсах, в диспетчері створюється таблиця, стовпці якої відповідають ресурси {R_i}, рядкам завдання з сформованої черги завдань. На перетині рядка і стовпця будемо ставити одиницю, якщо з порівняння $(\alpha_1^{R_i}, \alpha_2^{R_i}, ..., \alpha_q^{R_i})$ і $(\alpha_1^{З_i}, \alpha_2^{З_i}, ..., \alpha_q^{З_i})$ і випливає, що дане завдання може бути реалізоване на наявному вільному ресурсі і нуль в іншому випадку. Вирішуючи задачу про найменшому покритті (ЗНП), знаходимо мінімальне число ресурсів, на якому сформована черга завдань може бути виконана, і відправляємо завдання на рішення у виділені ресурси. Далі черга поповнюється новими завданнями, і процедура повторюється, ті завдання, для яких ми в таблиці маємо нульові рядки, оправляються назад в чергу і робиться повідомлення адміністратору про неможливість виконання даного завдання не на одному ресурсі системи.

Математичною моделлю роботи диспетчера що реалізує процедуру D є задача про найменшому покритті (ЗНП), яка може бути сформульована як задача лінійного булевого програмування виду.

$$L = \sum_{j=1}^{n} x_j \to \min \qquad (1)$$

при обмеженнях

$$\sum_{j=1}^{n} \beta_{ij} x_j \ge 1, \ \mathrm{i} = \overline{1, m}; \quad \beta_{ij} \in \{0,1\}; \ x_j \in \{0,1\}. \qquad (2)$$

Особливістю реалізації диспетчером процедури планування на основі рішення ЗНП є, те, що рішення ЗНП здійснюється постійно, з періодом визначеним часом збору інформації про стан ресурсів. Динамічна процедура D здійснює формуванням локальних черг, забезпечуючи максимальне завантаження кожного ресурсу. Крім розглянутої процедури планування на основі рішення ЗНП широко використовуються процедури FCFS (First Come First Served) - які розподіляє завдання за принципом «перший прийшов першим обслужили», тобто в порядку їх надходження (у порядку черги), вибираючи при цьому для завдання вільний і доступний ресурс, і централізований метод планування ресурсів в Грід [6]. Планувальник вибирає перше завдання з черги пулу і пробує помістити її в пакет завдань на один з ресурсів, на яких вона може бути вирішена. Якщо операція виконана успішно, то планувальник переходить до нового завдання, якщо ні - повертає його в пул, при цьому воно залишається першим в пулі і переходить до наступного завдання з черги.

Формалізація та постановка задачі при використанні як динамічної процедури завдання нелінійного булевого програмування (NLP) на основі групової вибірки з індивідуальною сегментацією. Спосіб групової вибірки з індивідуальною сегментацією це такий спосіб, при реалізації якого завдання, що знаходяться в черзі, розбиваються на підзавдання [1-3]. З черги отриманих підзавдань вибираються такі, які можуть бути вирішені на різних процесорах, і щоб сума їх пріоритетів була максимальною. У разі наявності рівнозначних підзавдань вибирають більш «старі». Підзавдань - це частина завдання, яка може бути вирішена на одному конкретному процесорі. Якщо завдання потребує використання R процесорів, то воно розбивається на R підзавдань. У цьому варіанті нам треба вибрати з черги як можна більшу кількість підзавдань, які звертаються до різних процесорам, і при цьому сума пріоритетів обраних підзавдань повинна бути максимальна. Слід зазначити, що прагнення до максимуму суми пріоритетів обраних підзавдань є головним критерієм

при виборі підзавдань з черги. Нехай $\{\vec{X}\}$ - множина всіх варіантів вибірки завдань з черги, $\vec{X}$ - один з варіантів вибірки завдань. Причому

$$\vec{X} = \{x_1, x_2, ..., x_p, ..., x_N\}, \quad (3)$$

де $p = \overline{1..N}$, N кількість завдань у черзі, x_p - булева змінна дорівнює 1 якщо завдання $З_p$ вибрано в цьому варіанті і 0, якщо не вибрано. $С_p$ - пріоритет завдання $З_p$.

Для оцінки способу вибірки завдань будемо використовувати коефіцієнт використання ресурсів $К_{ВР}$. Даний коефіцієнт показує, яка частина ресурсів, із загальної кількості ресурсів до яких звертаються стоять в черзі завдання, буде використана. Якщо $К_{ВР}$ = 0, то жоден ресурс не буде використаний, а якщо $К_{ВР}$ = 1, то використовуються всі ресурси, для яких існують завдання на даний момент. Але не завжди можливо вибрати такі завдання, щоб всі ресурси були задіяні. Тому перед нами стоїть завдання вибору такого способу вирішення черги завдань, при якому $К_{ВР}$ прагнуло б до 1. В загальному вигляді $К_{ВР}$ визначається таким чином:

$$K_{BP} = \frac{N_B}{N_O}, \quad (4)$$

де N_B - кількість ресурсів, які будуть задіяні при реалізації певної вибірки; $N_ч$ - кількість ресурсів, до яких існують завдання у черзі.

Але в даному виді $К_{ВР}$ не враховує пріоритети запитів, тому необхідно змінити коефіцієнт (4), для обліку пріоритету запитів. Але якщо враховувати пріоритети запитів, то уявлення коефіцієнта $К_{ВР}$ залежить від обраного способу вибірки. Нехай при груповій вибірці з сегментацією $З_{kg}$ - підзавдань завдання $З_k$ яке може бути вирішено на процесорі $Т_g$. $С_{kg}$ - пріоритет запиту $З_{kg}$. $\{\vec{X}\}$ - множина всіх варіантів вибору підзапитів з черги, $\vec{X}$ - один з варіантів вибору підзапитів. Причому $\vec{X} = \{x_{11}, x_{12}, ..., x_{kg}, ..., x_s\}$, де $k = \overline{1..p}$, $g = \overline{1..M}$, p - кількість запитів в черзі, M - кількість процесорів, x_{kg} булева змінна дорівнює 1 якщо відповідне підзавдань $З_{kg}$ вибрано в цьому варіанті і 0 якщо ні, тоді коефіцієнт (4) прийме вигляд:

$$K_{BP} = \frac{\sum_{j=1}^{p_1} C_{1j} S_1(C_n^1) + \sum_{j=1}^{p_2} C_{2j} S_2(C_n^2) + ... + \sum_{j=1}^{p_k} C_{kj} S_k(C_n^k) + ... + \sum_{j=1}^{p_n} C_{nj} S_n(C_n^n)}{\sum_{i=1}^{M} Y_i}, \quad (5)$$

де $Sr(C_n^r) = S_1 + S_2 + ... + S_{p_r}$ - сума всіх можливих поєднань творів змінних містять в кожному добуткові $Sr = X_p X_к ... X_m$ r різних змінних; $p_r = \frac{n!}{r!(n-r)!}$; C_{rj} - коефіцієнти, що стоять в добутках Sr, містять змінні r і визначають важливість вирішуваних завдань; Y_i - максимальна величина пріоритету завдання із завдань черги, що звертаються до ресурсу Ri,

I, відповідно, виходячи з (6) отримуємо функціонал (7):

$$F(x) = \sum_{j=1}^{p_1} C_{1j} S_1(C_n^1) + \sum_{j=1}^{p_2} C_{2j} S_2(C_n^2) + ... + \sum_{j=1}^{p_k} C_{kj} S_k(C_n^k) + ... + \sum_{j=1}^{p_n} C_{nj} S_n(C_n^n) \to \max \quad (6)$$

Ми отримуємо задачу нелінійного програмування з булевими змінними, з обмеженнями виду

$$\sum_{k=1}^{p} A_{kg} x_k \le B_g, \; g = \overline{1..M} \quad (7)$$

де $А_{kg}$ - булева змінна дорівнює 1 якщо $З_k$ використовує процесор $Т_g$ і 0 якщо немає; B_g - кількість ресурсів, яке можна використовувати, для виконання завдань $З_k$ в даний момент часу з безлічі M, виходячи з умови, що в будь-який момент часу будь-який процесор може бути використаний для рішення одного завдання. В якості методу розв'язання задачі (6,7) розробленої моделі використовувався ранговий метод вирішення даної задачі запропонований в [6] /

Порівняння стратегії *NLP* з *MC* і *FCFS* здійснювалося при наступних умовах:

1. ***Система планування*** організована у вигляді дворівневої структури, на першому рівні якої вибираються завдання, що підлягають плануванню, до них застосовується методи вирішення задач, вибрані як результат її вирішення завдання призначаються на доступні і вільні на момент розподілу ресурси і вирішуються на них під керуванням локального планувальника.

2. ***Метод планування*** на кожному кроці планування максимально завантажує мінімальна кількість вільних і доступних на момент планування ресурсів. Таким чином, для подальшого розподілу завдань черги кількість ресурсів для можливого призначення на них завдань ***буде максимальним.***

3. ***Алгоритм*** розв'язання задачі планування повинен мати малу тимчасову складність його реалізації для мінімізації часу, відведеного на процес планування завдань для їх вирішення на ресурсах.

4. ***Система планування*** використовує пакетну технологію «pull-push»: завдання, організовані у формі пакета (пулу завдань), вибираються з черги (pull), по мірі їх планування на ресурси вкладаються в пакет завдань на призначений ресурс (ресурси) і передаються на рішення (push) на цей ресурс (ресурси).

5. На моменти планування t_k завдання m є незалежними і n ресурсів системи є доступними і вільними.

Таким чином, ***мета дослідження*** - порівняти процедури планування *NLP*, *MC* і *FCFS* на основі метрик продуктивності роботи GRID-системи.

Постановка експерименту і метрики продуктивності роботи системи

В якості метрик продуктивності роботи моделі системи використовувалися середній час відповіді і коефіцієнт використання ресурсів.

Для даної системи час відповіді визначається таким чином:

$$t_{\text{відповіді}} = t_{\_\text{пул}} + t_{\_\text{пакет}} + t_{\_\text{розв'язання}},$$

де N - кількість завдань, що знаходяться в глобальній черзі.

Коефіцієнт використання ресурсів

$$K_{\text{ир}} = \sqrt[k]{K_{u1} K_{u2} \ldots K_{uk}}$$

де $K_{\text{Пі}} = \dfrac{P_{\Pi i}}{P_{\max}}$ - коефіцієнт використання *i*-го ресурсу; P_i - сумарний пріоритет завдань виконаних *i*-м ресурсом; *k*-кількість тактів за яке виконується вся черга завдань.

Коефіцієнт пріоритетності

$$K_{\text{пр}} = \sqrt[k]{K_{\Pi 1} K_{\Pi 2} \ldots K_{\Pi k}}$$

де $K_{\text{Пі}} = \dfrac{P_{\Pi i}}{P_{\max}}$ - коефіцієнт пріоритетності на *i*-му такті планування;

$P_{\text{Пі}}$ - сумарний пріоритет завдань розміщені в буфер на *i*-му такті планування;

$P_{\max}$ - максимальний пріоритет завдання в черзі.

Базова конструкція і принципи роботи моделі

Для опису конструкції і роботи пропонованої моделі використовуємо такі поняття і визначення. В якості базової архітектури моделі вибирається архітектура обчислювальної GRID-системи, що використовує ієрархічну структуру: на першому рівні знаходиться система планування ресурсів (кластерів) GRID-системи (планувальник першого рівня), на другому - локальному - рівні здійснюється локальне планування завдань на ресурсі (локальний планувальник), призначене для планування рішення призначених на даний ресурс завдань.

У дворівневій архітектурі GRID-системи пропонується використовувати наступні компоненти:

- глобальна черга, яка формується в міру надходження завдань користувачів для їх вирішення в GRID-систему; глобальна чергу характеризується розміром $L_{GlQuer,}$ - кількістю завдань, які надійшли на їх рішення в GRID-систему за час формування черги T_{GlQuer},;
- проміжний пул завдань, вибраних з глобальної черги, що має розмір L_{pool} ($L_{pool} \le L_{GlQuer}$). Розмір пулу може визначатися, наприклад, періодичністю процесу планування: днями, тижнями і т.д.;
- пакет завдань, що формується як результат обробки завдань пулу на основі рішення задачі про найменшому покритті, що має певний розмір ***Lpacket***;
- ресурси R_i представляють собою кластери GRID-системи, на яких будуть вирішуватися сплановані планувальником першого рівня завдання.

Послідовність обробки завдань, що знаходяться в глобальній черзі, представляє циклічний процес планування, який можна представити у вигляді динамічного алгоритму планування базується на основі використання процедур *NLP* з *MC* і *FCFS*.

Для перевірки ефективності запропонованого підходу до планування ресурсів розроблена програмна реалізація моделі, що реалізує імітаційну дискретно-подієву модель функціонування гетерогенної GRID-системи, що включає наступні функціональні елементи:

1. Глобальна черга завдань - завдання (заявки, вимоги, додатки), які подаються на систему для моделювання процесу роботи гетерогенної середовища.

2. Проміжний пул завдань - буфер системи, в який в порядку надходження завантажуються завдання з глобальної черги, і здійснюється планування ресурсів.

3. Планувальник - програмний модуль, який розподіляє на основі обраного методу планування завдання, що знаходяться в пулі, на локальні ресурси.

4. Модель завдань - визначається складністю завдань, задається в тактах - більш складне завдання вимагає для його виконання більшої кількості тактів.

5. Пакет завдань - набір завдань направляються на ті ресурси, на яких дані типи завдань можуть бути виконані.

6. Ресурси - елементи, що моделюють процес виконання завдань, які були на них розподілені, продуктивність яких теж визначається числом тактів необхідних для звільнення ресурсів.

7. Затримка - характеризує комунікаційну затримку даних, що передаються з пакета завдань для виконання на кластер.

8. Модель часу – в якості моделі часу була обрана умовна одиниця часу роботи системи - такт, який можна розглядати як тривалість інструкцій або команд в обчислювальній системі.

Один такт включає виконання в моделі трьох операцій:

- **Операції №1** - розміщення вхідних завдань в пул;
- **Операція №2** - розподіл задач в черзі пулу між ресурсами і повернення не помістившихся на ресурси завдань назад в пул;
- **Операція №3** - імітація вирішення завдань ресурсами, на які вони були розподілені.

При моделюванні досліджувалися залежності часу виконання всіх завдань глобальної черги в залежності від кількості завдань у черзі при різних складнощах розв'язуваних завдань і різної продуктивності ресурсів, а також залежність коефіцієнта використання ресурсів гетерогенної GRID-системи від кількості розв'язуваних завдань. Характеристики наведені на рис.2-5 отримані при числі ресурсів використовуваних в гетерогенної GRID-системі рівному 10 з довірчою ймовірністю рівної 0,95, при цьому всі параметри GRID-системи змінювалися за законом Пуассона.

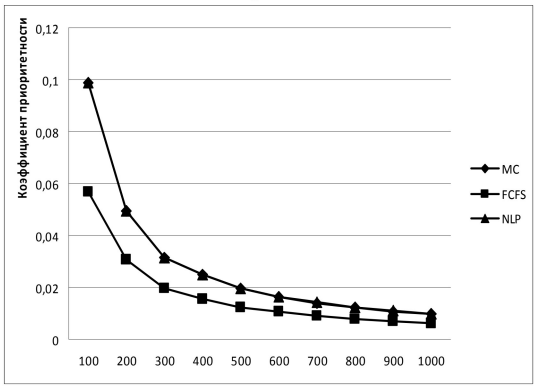

Рисунок 2 - Залежність коефіцієнта пріоритетності від числа завдань у черзі

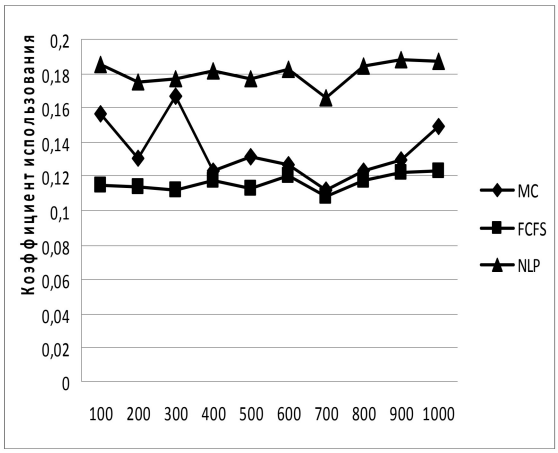

Рисунок 3 - Залежність коефіцієнта використання ресурсів від числа завдань у черзі.

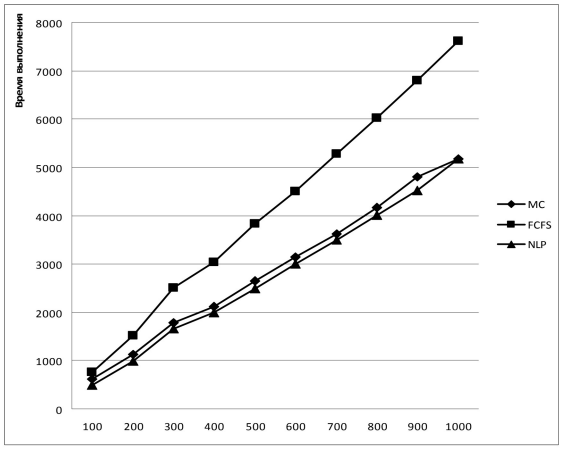

Рисунок 4 - Залежність часу виконання всієї черги завдання від числа завдань у черзі

Рисунок 5 - Зміни сумарного пріоритету виконаних завдань на різних тактах планування

З рисунків 2-4 видно, що при використанні для планування процедури, базуються на вирішенні задачі нелінійного програмування (6,7) коефіцієнт використання і коефіцієнт пріоритетності вище ніж у процедур планування на основі рішення ЗНП (1,2) і процедур *FCFS*. При цьому оперативність виконання черги завдань вище у процедури *NLP*.

Висновок

Обчислювальні експерименти, проведені з використанням програмної реалізації розглянутої моделі, підтвердили ефективність запропонованої процедури планування *NLP* порівняно з процедурами, заснованими на вирішенні ЗНП і процедур *FCFS*.

Яким представляється розвиток подальших робіт з проблематики GRID-у?

Перш за все це розробка типових рішень. GRID-и повинні створюватися зацікавленими в них організаціями, а роль системних фахівців полягає в тому, щоб запропонувати технологічні рішення для типових завдань і відбір базових програмних засобів. Так, у справі створення інфраструктури хостингу

основне - рішення двох проблем.

Перше - розробка загально GRID-овскіх серверів, в контейнерах яких можуть бути встановлені інтеграційні і підтримуючі GRID-служби. Друге – розробка шлюзових служб, через які ресурси підключаються до GRID. Створення ж ресурсної складової інфраструктури, яке часом виступає головним завданням, має бути залишено творцям конкретних GRID. Приблизно таким же бачиться вирішення завдання розробки розподілених додатків. Якщо говорити про масове застосування GRID-у, то додатки повинні бути предметно-орієнтованими, і їх розробка повинна здійснюватися прикладними фахівцями, а завданням системні фахівці повинні забезпечити їх типовими рішеннями та інструментальними засобами. Розвиток досліджень робіт по GRID-комп'ютингу протікає надзвичайно енергійно і характеризується постійною появою нових концепцій, архітектурних рішень, технологій і розробок, так що згаданий вище фундамент постійно розширюється і можливості з побудови ефективних безпечних систем управління залізничним транспортом на основі GRID технологій.

Література

1. *LCG Middleware.* http://lcg.web.cern.ch/LCG/activities/middleware.htm
2. *Workload Management System User and Reference Guide.* https://edms.cern.ch/file/572489/1/EGEE-JRA1-TEC-572489-WMS-guide-v0-2.pdf
3. . *R-GMA C User and Reference Guide.* https://edms.cern.ch/file/503615/1.5/EGEE-JRA1-TEC-503615-v1.4.pdf
4. *DataGrid Project Documentation.* http://marianne.in2p3.fr/datagrid/documentation/
5. *Condor Project*: http://www.cs.wisc.edu/condor
6. *Пономаренко В.С.* Методы и модели планирования ресурсов в ГРИД--системах: монография / В.С. Пономаренко, С.В. Листровой, С.В. Минухин, С.В. Знахур. – Харьков: ИД «ИНЖЭК», 2008. – 408 с.

Листровой С. В., Лаврик С. Е., Листровая Е. С. (УкрГАЖТ)

О ПОДХОДАХ К ПОСТРОЕНИЮ ПОЛИНОМИАЛЬНЫХ АЛГОРИТМОВ РЕШЕНИЯ ЗАДАЧ ДИСКРЕТНОЙ ОПТИМИЗАЦИИ ОТНОСЯЩИХСЯ К КЛАССУ NP-ПОЛНЫХ ЗАДАЧ

Рассматривается возможность построения полиномиальных алгоритмов решения задач определения максимальных независимых множеств и решения *SAT*-задач. Во многих прикладных задачах синтеза и анализа вычислительных систем и сетей и разработки специального математического обеспечения для их функционирования требуется найти в конечном множестве объектов максимальную систему объектов, попарно не связанных друг с другом, или же выбрать минимальную систему объектов, связанных со всеми другими. Формулировки подобных задач на языке теории графов приводят к понятиям независимости и покрытия. Задача *SAT* это задача определения разрешимости булевой формулы. Формула называется разрешимой, если для нее существует выполняющий ее набор переменных, то есть набор значений всех входящих в формулу переменных, на которых формула истинна. В русскоязычном варианте она известна как задача «выполнимость» (ВЫП). Данная задача имеет важное прикладное значение: при верификации программного и технического обеспечения современных ЭВМ; при проектировании ПЛИС, при решении задач автоматизации доказательств, связанных с проверкой противоречивости множества дизъюнктов в исчислении высказываний. Она так же находит широкое применение в криптографическом анализе, поскольку алгоритмы шифрования можно рассматривать в терминах КНФ (конъюнктивная нормальная форма) и интерпретировать задачу криптографического анализа, как задачу нахождения решающего набора, где решающим набором является секретный ключ. Задача *SAT* имеет важное значение в системах автоматической проверки доказательств, где формулой называют набор клозов (clause), под которыми понимается дизъюнкция некоторого количества литералов – перемеренных X и $\overline{X}$. Большое значение, данная задача имеет при выяснении выполнимости схем CIRCUIT-SAT (circuit-satisfiability problem).

Ломотько Д. В., Бронза С. Д., Овчієв М. Ж. (УкрДАЗТ)

РОЗПОДІЛ ІМОВІРНІСТІ СТАНІВ СИСТЕМИ ОБОРОТУ ВАГОНІВ НА ЗАЛІЗНИЧНОМУ ВУЗЛІ. ЗАГАЛЬНЕ РІШЕННЯ

У доповіді розглянуто систему масового обслуговування (СМО) типу вантажний залізничний вузол (ВЗВ), (яка розглядається як марковський ланцюг) з метою обчислення розподілу імовірності станів обігу вагонів. Отримано загальне рішення поставленої задачі. Обчислені імовірності знаходження вагонів в будь-якому стані в довільний момент часу, наведено два приклада моделювання.