УДК-004.7

Марич Володимир, Волокита Артем

СПОСІБ ПЛАНУВАННЯ ПЕРЕВІРКИ РЕЗУЛЬТАТІВ ВОЛОНТЕРСЬКИХ ОБЧИСЛЕНЬ ДЛЯ ЗАДАЧ РЕАЛЬНОГО ЧАСУ

METHODS OF SCHEDULING FOR CHECKING THE RESULTS OF VOLUNTEER COMPUTING FOR REAL-TIME TASKS.

У статті представлено спосіб планування перевірки результатів волонтерських розподілених обчислень, який за рахунок оцінки обсягу перевірок на основі скорингової моделі з алгоритму планування earliestdeadlinefirst, дозволяє використовувати волонтерські обчислення для задач реального часу.

Ключові слова: волонтерські обчислення, планувальник задач, скоринг, грід система, реальний час.

Рис.: 1. Бібл.: 10.

The article presents a method for planning the verification of the results of volunteer distributed computing, which, by estimating the amount of rechecks based on a scoring model with a planning algorithm earliest deadline first, it allows you to use volunteer calculations for real-time tasks.

Key words:volunteer computing, task scheduler, scoring, grid system, real-time.

Fig.: 1. Bibl.: 10.

Актуальність теми дослідження. З кожним роком обсяг і складність обчислювальних задач монотонно зростають, можливостей персональних комп’ютерів не вистачає, і тому багато хто звертається до компаній, що володіють суперкомп’ютерами, або обраховують свої задачі в хмарах. Дані послуги є дуже дорогими, тому в останній час велику популярність набирають системи добровільних розподілених обчислень. Переважна більшість цих систем безкоштовні. В якості обчислювальної одиниці виступають персональні комп’ютери, або ноутбуки будь-якої людини, що добровільно надала доступ до своїх ресурсів. Це особливо турбує дослідників, чиї погляди розвиваються швидше, ніж їх комп'ютери. Томудослідження протікають дуже і дуже повільно, через брак ресурсів або коштів, необхідних для виконання масивних обчислювальних проектів.

Дані системи побудовані на мультиагентній структурі. Така структура дозволяє за невеликий проміжок часу організувати розподіленні обчислення на основі звичайної комп’ютерної мережі. Яскравим прикладом такої програмної системи є BOINC [1].

Постановка проблеми. При організації розподілених обчислень на основі звичайних мережевих комп'ютерів можуть виникати загрози безпеки обчислювальних процесів, а саме відмова обладнання або несанкціоновані дії зловмисників, що спрямовані на блокування вузлів обчислювальної системи або дискредитацію результатів роботи розподіленого обчислювального процесу.

Перша загроза не є надто суттєвою, бо у разі втрати доступу до ресурсів однієї, або декількох обчислювальних машин, система перерозподілить задачі

між іншими. Тому основною проблемою виступає – недостовірність отриманих результатів обчислень.

Є лише єдина можливість впевнитись в правильності отриманої інформації – це повторний обрахунок задачі на “перевірених” комп’ютерах. Вони називаються агентами системи. Однак ресурс агентів і час обчислення задачі обмежені. Тому виникає питання оцінки обсягу перевірки результатів.

Аналіз останніх досліджень і публікацій. У статті [2] розглядається проблема оптимізації вибіркових перевірок для забезпечення надійності грід систем, в яких деякі ненадійні вузли (диверсанти) можуть виконувати атаки на системи.

У статті [3] автори узагальнюють вибіркову перевірку шляхом введення ідеї перевірки дефектів. Це узагальнення дозволяє гарантувати коректність обчислень в ситуації, коли вибіркова перевірка не є повністю надійною.

У статті [4] описується грід система, яка дозволяє здійснювати обчислення на віддаленій, а також потенційно небезпечній, системі для перевірки прогресу та правильності виконання обчислень.

Виділення недосліджених частин загальної проблеми. Виходячи із недоліків структури добровільних розподілених обчислень, а саме можливості дискредитації результатів з боку зловмисників, та обмеженості ресурсів агентів, виникає питання створення способу планування перевірки результатів.

Постановка завдання. Створити спосіб планування перевірки результатів волонтерських обчислень в реальному часі, що включає визначення обсягу перевірки та їх порядок.

Викладення основного матеріалу. Мультиагентна система [5] являє собою множину агентів Aу вигляді перевірених надійних комп’ютерних модулів агентів $\{a_1, a_2, \ldots, a_n\} \subseteq A$. Множина Aнакладається на множину $\{p_1, p_2, \ldots, p_j\} \subseteq P$мережевих комп'ютерів ($P=A$) так, що агент a_i закріплюються за відповідним p_i комп'ютері мережі. Агенти взаємодіють через сервер.

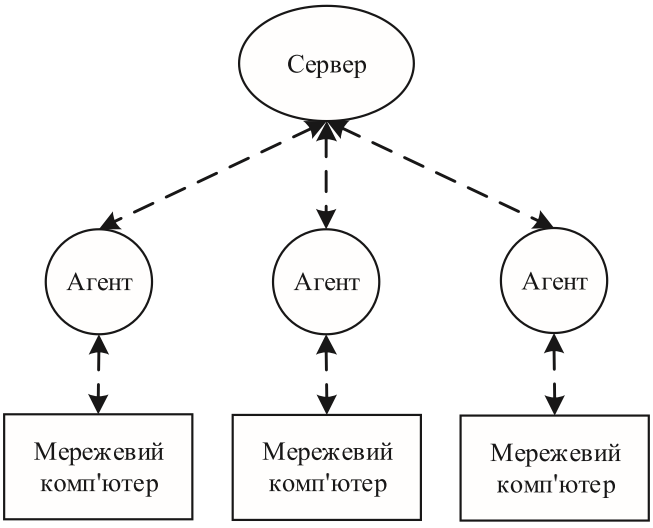

Рис.1 Структура мультиагентної системи

Вся мультиагентна система, керуючи комп'ютерами p_i, організовує систему розподілених обчислень для вирішення всієї множини T завдань $\{t_1, t_2, \ldots, t_n\} \subseteq T$.

Кожен модуль агента керує ресурсами комп'ютера і стежить за виконанням завдання. У свою чергу, кожен комп'ютер працює незалежно від інших комп'ютерів.

Алгоритм агента функціонує наступним чином. На початку організації розподілених обчислень в комп'ютерній мережі $\{p_1, p_2, \ldots, p_n\} \subseteq P$ знаходяться, керуючі їх роботою, агенти $\{a_1, a_2, \ldots, a_n\} \subseteq A$. На першому етапі агент a_i, отримує основну інформацію для організації розподілених обчислень. Вона включає в себе обчислювальну задачу t_iі її частину, яку агент повинен виконати. Після отримання агентом задачі і загальної інформації про систему він ініціює на своєму комп'ютері обчислювальний процес для виконання t_i. Після виконання кожного рішення агент розсилає результат усім іншим агентам.

Очевидно, що використання окремого агента для контролю лише одного обчислювального модуля є не раціональним. Тому кожен агент має в своєму розпорядженні декілька модулів.

Оскільки система складається з добровільно наданих комп’ютерів p_i - існує ймовірність втручання зловмисників, для дискредитації результатів. Тому одним із найголовніших пунктів, що входять в обов’язки роботи агента, це перевірка валідності обчислень.

Кожна мультиагентна система, відповідно до своїх ресурсів, може вирішувати декілька задач. Для кожної задачі система вираховує кінцевий час вирішення обчислення. Для підтримки цілісності системи, результати цих задач повинні бути надані вчасно. Цю проблему також повинні вирішувати агенти у реальному часі.

Виходячи з цих потреб, та беручи до уваги, що ресурси агентного модуля обмеженні, постає питання оцінки обсягу результату який потрібно перевірити. Для кожного результату, наданими окремим обчислювальним модулем, його частина для повторного обчислення, та позиція в черзі на перевірку в агенті буде різною.

Для впевненості у достовірності отриманої інформації слід враховувати велику кількість факторів, які на пряму впливають на обсяг перевірки результату та на пріоритет в черзі. І з основних можна виділити:

- кінцевий час виконання задачі;
- складність та важливість задачі;
- обсяг отриманого результату;
- час витрачений на розв’язок;
- довіра до обчислюваного модуля.

Перші два фактори напряму задаються системою. Третій і четвертий отримуємо після закінчення виконання обчислень. Останній включає в себе комплекс факторів, і теж вираховується системою.

Високий рівень довіри може мати об'єкт, який не є джерелом деструктивних впливів. Низьким рівнем довіри володіють об'єкти, які щойно з'явилися в системі, раніше невідомі об'єкти, тобто об'єкти, які є або можуть завдавати шкоди системі.

Тобто рівень довіри до об'єкта – це ймовірність того, що об'єкт не є джерелом деструктивного впливу. Рівень довіри в мультиагентної системи обчислюється різними методами, де основною ідеєю виступає сумарна оцінка для кожного обчислювального модуля від кожного агента [6-8].

Всі ці фактори є різнотипними, тому для оцінки їх впливу пропонується використовувати скорингову модель.

ЇЇ суть [9] полягає в тому, що кожному параметру, що характеризує об’єкт, надається реальна оцінка в балах. У спрощеному вигляді скорингова модель – це зважена сума визначених характеристик. Така методика є універсальною і може застосовуватись для будь-яких об’єктів. За допомогою скорингу на основі виділених факторів можна визначити, наскільки велика ймовірність того, що конкретний об’єкт може вчинити ті чи інші дії. Данна модель вже давно використовується у кредитній системі. Загальний вигляд скорингової формули:

$$S = \sum_i^n k_i * x_i \quad (1)$$

S – сумарний бал;
k_i – коефіцієнт;
x_i – бал окремого фактору.

Для кожного окремого фактора x_i є своя верхня оцінка. Якщо ні, то слід використовувати монотонно зростаючі обмежені зверху і знизу функції $f(x_i)$. Відповідно до характеру прояву фактору можна використовувати ряд наступних функції: функція експоненціального розподілу; функція Гамма-розподілу; функція розподілу Вейбула. Всі вони монотонно зростають від 0 до 1.

Завдяки тому, що під час отримання локальних максимумів окремих факторів, існує загальний максимум, можна точно визначити обсяг отриманих даних який потрібно перевірити. Формула має наступний вигляд:

$$n = (1 - {}^{S}/_{S_{max}}) * N \quad (2)$$

S_{max} – максимальний можливий бал;
N – розмір отриманого результату;
n – розмір результату який потребує перевірки.

Найголовнішими факторами для обчислення обсягу перевірки є довіра та складність задачі. Якщо складність для всіх обчислювальних модулів однакова, бо вони вирішують одну задачу, то ступінь довіри до кожного – індивідуальна. Тому при правильному виборі коефіцієнтів та формул, даний підхід надає змогу швидко та з високою точність перевіряти результати отриманні від добровільних обчислювальних модулів.

Частини результатів, що потребують повторного обчислення, для комп’ютерів з високим балом довіри будуть менші, і навпаки. Також можна додатково ввести обмеження, що розмір і важливість задач, які надаються для обробки, напряму залежить від ступеню довіри. Це дозволить запобігти втручанню зловмисників, оскільки для дискредитації серйозних розмірів результату всіє задачі, вони потребують підвищення рівню довірі, що в свою чергу залежить від кількості правильно обчислених раніше задач.

Такі фактори як: кінцевий час виконання, складність і важливість задачі, та час витрачений на розв'язок, використовуються під час обчислення пріоритету задачі в черзі. Для його підрахування можна також використовувати скорингову модель, за формулою (1). Чим більший бал, тим ближчий номер задачі в черзі агента, на виконання повторного обчислення.

Процес вибору наступного результату за пріоритетом, подібний до планування процесів на виконання у комп'ютерній операційній системі. Існує два основних підходи статичний або динамічний. Статичне планування обчислює виконання замовлень перед початком виконання. Це вимагає знання характеристик задач, і створює невеликі накладні витрати. Динамічне планування приймає рішення під час виконання процесів. Це дозволяє створювати більш гнучку систему, але потребує додаткових витрат.

Мультиагентна система відноситься до систем, що працюють в режимі реального часу, тому доцільніше використання алгоритмів з динамічним планування. Враховуючи характеристики системи слід розглядати earliestdeadlinefirst (EDF) [10]. Він працює наступним чином: при настанні дії планування (задача завершена, встановлена нова задача і т. д.), планувальник шукає найближчий до крайнього часу її виконання завдання, і цей процес призначається для виконання.

Планування за найближчим терміном завершення є оптимальним переважно для одно-процесорних систем, в тому відношенні, що якщо існує можливість виконати набір незалежних завдань, кожен з яких характеризується часом настання, вимогою виконання та крайнім терміном завершення, так, щоб гарантувати виконання всіх задач до крайнього терміну.

В даній моделі під час надання пріоритетів агент враховує фактори, які впливають на організацію черги в EDF-алгоритмі. Пріоритети слід обраховувати після кожного вибору задачі із черги, для підтримання динамічності системи.

Висновки. Проаналізувавши вразливість добровільних обчислень, було розроблено спосіб планування перевірки результатів обчислень в реальному часі. В якості оцінки обсягу використовується скорингова модель. Планування базується на алгоритмі EDF.

В подальших дослідженнях авторами плануються розглянути різні типи задача реального часу, в тому числі розпізнавання образів і block-chain обчислення., та запропонувати більш ефективні та захищенні способи перевірок для таких задача.

Список використаних джерел

1. Anderson, David P. Boinc: A system for public-resource computing and storage. //5th IEEE/ACM International Workshop on Grid Computing – IEEE Computer Society – 2004.

2. WatanabeK. etal. OptimalSpot-CheckingforDelayedAttackonDesktopGrid Systems //ComputerModellingandSimulation (UKSim), 2013 UKSim 15th International Conference on. – IEEE, 2013. – С. 600-605.

3. Fukushi M, Watanabe K. Generalized spot-checking for reliable volunteer computing //IEICE TRANSACTIONS on Information and Systems. – 2010. – Т. 93. – №. 12. – С. 3164-3172.

4. Yang S. et al. Trust but verify: monitoring remotely executing programs for progress and correctness //Proceedings of the tenth ACM SIGPLAN symposium on Principles and practice of parallel programming. – ACM, 2005. – С. 196-205.

5. Ховансков С. А., Литвиненко В. А., Хованскова В. С. Алгоритм организации безопасных распределенных вычислений на основе многоагентной системы //Известия Южного федерального университета. Технические науки. – 2016. – №. 10 (183)

6. Губанов Д. А. Обзор онлайновых систем репутации/доверия. – 2009.

7. Абрамов Е. С., Басан Е. С. Разработка модели защищенной кластерной беспроводной сенсорной сети //Известия Южного федерального университета. Технические науки. – 2013. – №. 12 (149).

8. Зикратов И. А. и др. Построение модели доверия и репутации к объектам мультиагентных робототехнических систем с децентрализованным управлением //Научно-технический вестник информационных технологий, механики и оптики. – 2014. – №. 3 (91).

9. Волик, Н. Г. Скоринг як експертний метод оцінювання кредитного ризику комерційного банку при споживчому кредитуванні //Вісник Запорізького національного університету: Економічні науки.–Запоріжжя – 2008. – ЗНУ 1 (40-44).

10. NAGER S. K., GILL N. S. Comparative Study of RM and EDF Scheduling Algorithm in Real Time Multiprocessor Environment. – 2017.

ДОВІДКА ПРО АВТОРІВ

Марич Володимир Васильович – студент 4-го курсу, Факультету інформатики та обчислювальної техніки, Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського».

Marych Volodymyr – student, Faculty of Informatics and Computer Science, National Technical University of Ukraine “Igor Sikorsky Kyiv Polytechnic Institute”.

E-mail: 777Titan111@gmail.com

Волокита Артем Миколайович – доцент, кафедра обчислювальної техніки, Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського».

Volokyta Artem – associate professor, Department of Computer Engineering, National Technical University of Ukraine “Igor Sikorsky Kyiv Polytechnic Institute”.

E-mail: artem.volokita@kpi.ua

Marych Volodymyr,
Volokyta Artem

METHODS OF SCHEDULING FOR CHECKING THE RESULTS OF VOLUNTEER COMPUTING FOR REAL-TIME TASKS.

Relevance of research topic. Every year, the amount and complexity of computing tasks monotonously increases, the capabilities of personal computers are not enough, and so many people turn to companies that have supercomputers, or calculate their tasks in the clouds. These services are very expensive. Therefore, in recent times, voluntary distributed computing systems are becoming increasingly popular. A good example of such a software system is BOINC.

Target setting. In the organization of distributed computing on the basis of original network computers, there may be security threats to performing computational processes, namely, hardware failure or unauthorized intruder actions. Resource agents and time calculations are limited. Therefore, there is a question of assessing the extent of verifying the results.

Actual scientific researches and issues analysis. Over the past years there are more articles devoted to it problems optimize sample checks to ensure the reliability of grid systems, in which some unreliable nodes (saboteurs) can perform attacks on systems.

Uninvestigated parts of general matters defining. Based on the shortcomings of voluntary distributed computing structure, namely the possibilities of discrediting the results from intruders, and agents of limited resources, the question arises about how to plan the results check.

The research objective. Create a method of planning real-time volunteer results checking, which includes determining amount of rechecks and their order.

The statement of basic materials. The analysis of the multiagent system is carried out. The vulnerable aspects of this model were identified. A method for scheduling a results check based on the scorecard model and EDF algorithm is described. The main factors that influence the amount of the result check and the calculation of the priority are considered.

Conclusions. After analyzing the vulnerability of voluntary computing, a method for planning the verification of the results of computations in real time was developed.