e-ISSN 1607-3274 Радіоелектроніка, інформатика, управління. 2020. № 2
p-ISSN 2313-688X Radio Electronics, Computer Science, Control. 2020. № 2

УДК 004.75

ВИКОРИСТАННЯ СЕСІЙНИХ МЕТРИК ПРИ ПЛАНУВАННІ ЗАВДАНЬ В СИСТЕМАХ ВОЛОНТЕРСЬКИХ ОБЧИСЛЕНЬ У БРАУЗЕРІ

Рибачок Н. А. – канд. техн. наук, старший викладач кафедри програмного забезпечення комп'ютерних систем Національного технічного університету України «Київський політехнічний інститут імені Ігоря Сікорського», Київ, Україна.

Орос Б. Б. – магістрант кафедри програмного забезпечення комп'ютерних систем Національного технічного університету України «Київський політехнічний інститут імені Ігоря Сікорського», Київ, Україна.

АНОТАЦІЯ

Актуальність. Дослідження присвячено розробці адаптивного методу планування завдань у системах розподілених волонтерських обчислень у браузері. Це дає можливість видавати кожному волонтеру пакет завдань для обчислень в залежності від характеристик його обчислювального процесу. При цьому продуктивніші користувачі будуть отримувати більший обсяг завдань, зменшуючи навантаження на сервер та сприяючи масштабуванню обчислень; менш продуктивні клієнти будуть отримувати менший обсяг завдань, вирішуючи проблему надлишкових блокувань вхідних даних.

Мета. Метою роботи є підвищення ефективності систем розподілених волонтерських обчислень у браузері шляхом використання сесійних метрик при плануванні завдань.

Метод. У роботі описано сесійні метрики, які визначають особливості поведінки користувачів-волонтерів. На їх основі введено показники, які характеризують перебіг обчислювального процесу на системах користувачів. Запропоновано формулу для обчислення обсягу пакету завдань для видачі клієнту, яка враховує продуктивність системи клієнта та тенденції протікання процесів на його системі. На основі існуючої системи волонтерських обчислень виконано моделювання процесу проведення обчислень із використанням різних методів розрахунку обсягу пакетів завдань. Для оцінки ефективності таких систем використано сумарну кількість звернень користувачів до серверної частини та сумарну кількість обчислених результатів.

Результати. Запропонований підхід застосовано в існуючій системі проведення волонтерський обчислень. Порівняння результатів роботи запропоновано методу із існуючими показало зменшення навантаження на сервер та пришвидшення отримання результатів обчислень при використанні сесійних метрик.

Висновки. Проведене моделювання підтвердило зручність та ефективність використання сесійних метрик при плануванні завдань. Представлений підхід забезпечує можливість масштабування систем розподілених волонтерських обчислень. Перспективи подальших досліджень полягають у використанні статистичних даних користувача та інформації від його браузеру для обчислення обсягів даних, що передаються клієнту.

КЛЮЧОВІ СЛОВА: розподілені волонтерські обчислення у браузері, планування завдань, пакет завдань, розподіл завдань, сесійні метрики.

АБРЕВІАТУРИ

ВО – волонтерські обчислення;

BBVC – browser-based volunteer computing.

НОМЕНКЛАТУРА

$\varepsilon_{l,k-1}$ – коефіцієнт для визначення продуктивності системи волонтера l під час звернення k–1;

$a_{l,k-1}$ – коефіцієнт, який показує зміну продуктивності системи волонтера l під час звернення k–1;

$D = \{D_1, D_2, ..., D_N\}$ – множина вхідних даних;

d_i – i-й проміжок вхідних даних;

$f(D)$ – функція для проведення обчислень;

$k(l)$ – встановлений волонтером l рівень навантаження на систему;

m – набір метрик, які характеризують поведінку волонтера під час звернення;

$M(m)$ – набір сесійних метрик;

$r_{l,k}$ – обсяг пакету завдань, що наданий волонтеру l під час звернення k,

$r_{l,k}$` – корегований обсяг пакету завдань, що наданий волонтеру l під час звернення k;

r_0 – початковий обсяг пакетів завдань;

$r_{\max}$ – максимальний обсяг пакетів завдань;

$r_{\min}$ – мінімальний обсяг пакетів завдань;

$R_l(k-1)$ – сумарна кількість обчислених проміжків даних за сесію із k–1-их звернень волонтером l;

$t_{l,i}^{\ j}$ – тривалість обчислення проміжку j із пакету завдань i волонтером l;

$t_{l,i}$ – тривалість обчислення пакету завдань i волонтером l;

$T_l(k-1)$ – сумарна тривалість обчислення проміжків даних за сесію із k–1-их звернень волонтером l;

$Q(M)$ – набір характеристик обчислювального процесу.

ВСТУП

Волонтерські обчислення є ідеєю розподілених обчислень, в якій користувачі віддають частину своїх обчислювальних ресурсів задля масштабних обчислювальних проектів, щоб зменшити їх вартість та пришвидшити отримання результату [1].

DOI 10.15588/1607-3274-2020-2-16

Розподілені волонтерські обчислення за допомогою браузеру є доволі перспективним рішенням, враховуючи поширеність пристроїв з підключенням до мережі Інтернет, об'єднані потужності яких можуть створити конкуренцію датацентрам [2]. Для того, щоб прийняти учать у таких обчисленнях, волонтерам не потрібно встановлювати спеціальне програмне забезпечення, достатньо лише перейти за певною адресою в браузері та зберігати сторінку відкритою [1].

Волонтерські системи мають клієнт-серверну архітектуру. Їх особливістю є те, що всі обчислення результатів відбуваються на стороні клієнта, а обов'язками сервера є планування завдань та агрегація результатів обчислень [3].

Планування завдань відіграє важливу роль в роботі таких систем, оскільки суттєво впливає на продуктивність і серверної, і клієнтської частин. Воно повинно сприяти ефективному використанню обчислювальних потужностей, які надані волонтерами [4].

Планування завдань – це складна робота, навіть для випадків з повною інформацією, а для реальних проектів воно стає ще складнішим [5]. Системи, які орієнтовані на проведення волонтерських обчислень через браузер (BBVC-системи), функціонують при дуже високій гетерогенності клієнтів. Умови виконання обчислень у таких системах є динамічними: змін можуть зазнавати як вхідні дані (наприклад, коли нові дані постійно надходять в систему), так і клієнти (наприклад, клієнти стають недоступними чи приєднуються нові) [6].

Задачу планування завдань можна розбити на два етапи: розподіл вхідних даних на проміжки та видача завдань для обчислень волонтерам [1]. Для підвищення ефективності при видачі завдань проміжки вхідних даних за звичай групуються у пакети [7]. Задача планування завдань у BBVC-системах найчастіше зводиться саме до задачі визначення обсягу пакетів завдань для передачі волонтерам.

У багатьох існуючих BBVC-системах для планування завдань застосовують підхід на основі надання однакової кількості проміжків [4], [8], [9]. Є реалізація, в якій продуктивнішим клієнтам надається більший обсяг даних для обчислень [10], адже це пришвидшує обрахунок та підвищує ефективність використання виділених ресурсів. Наведений у роботі [1] «адаптивним» алгоритм планування завдань полягає у наданні волонтеру пакету завдань більшого розміру при збільшенні тривалості його сесії обчислення. Але наведені підходи і алгоритми не враховують, що обчислювальні процеси на системах волонтерів мають складний динамічний характер і можуть як пришвидшуватися, так і сповільнюватися.

Якщо надавати клієнтам, які мають продуктивніші системи (швидше повертають обраховані результати), більший обсяг даних для обчислень, це також зменшить кількість комунікацій між сервером та клієнтами. І, як наслідок, сприятиме масштабуванню системи [11].

З іншого боку, якщо зменшувати обсяг даних для повільніших клієнтів, це вирішить проблему надлишкових блокувань, коли вхідні дані, що видані клієнтові для обчислень, блокуються сервером і не підлягають видачі іншим волонтерам.

В роботі пропонується планувати завдання для кожного волонтера на основі його сесійних метрик. Це надасть можливість надавати обсяг даних для обчислень, який підлаштований під особливості обчислювального процесу кожного користувача. Результатом буде зменшення навантаження на серверну частину у вигляді надлишкових запитів та блокувань.

Об'єктом дослідження є процес планування завдань.

Предметом дослідження є методи та алгоритми планування завдань.

Метою роботи є підвищення ефективності обчислень у BBVC-системах завдяки врахуванню характеристик обчислювального процесу користувача, отриманих на основі сесійних метрик, при плануванні завдань.

Для досягнення мети в роботі необхідно розв'язати наступні задачі:

– провести аналіз існуючих методів та алгоритмів планування завдань;

– формалізувати поведінкові особливості користувачів у вигляді метрик та ввести показники для оцінки характеру обчислюваного процесу їх системи;

– запропонувати формулу визначення обсягу пакету завдань на основі сесійних метрик та сформулювати відповідний метод планування завдань;

– провести моделювання роботи BBVC-системи із використанням відомих та запропонованого методу.

Методи дослідження – формалізація сесійних поведінкових особливостей користувача у вигляді метрик із подальшим їх використанням у формулах визначення обсягу пакету завдань.

1 ПОСТАНОВКА ЗАДАЧІ

Єдиний загальноприйнятий опис задачі планування завдань в BBVC-системах відсутній.

Розглянемо цю задачу у наступній постановці.

Нехай дана деяка функція $f(D)$ та множина вхідних даних $D = \{D_1, D_2, ..., D_N\}$, для кожного елементу якої необхідно обчислити відповідне значення функції. Перед проведенням обчислень множина вхідних даних ділиться на проміжки d_i, які мають однакову потужність та не перетинаються.

Кожного разу k, коли волонтер l звертається до сервера, той надає для обчислення пакет завдань – перші із необчислених проміжків вхідних даних обсягом $r_{l,k}$.

Необхідно визначити набори поведінкових метрик m, сесійних метрик $M(m)$, характеристик обчислювального процесу $Q(M)$ та на їх основі сформулювати метод визначення обсягу пакету завдань $r_{l,k}(Q)$.

DOI 10.15588/1607-3274-2020-2-16

2 ОГЛЯД ЛІТЕРАТУРИ

У [12] запропоновано наступний поділ методів планування завдань:

– «наївні», що не враховують історію обчислень;

– з використанням інформації про обчислювальні вузли, включаючи збір статистики обчислень;

– адаптивні, що змінюють параметри роботи «на льоту».

До першої групи відноситься метод надання волонтерам однакової кількості проміжків. Він є найпростішим методом визначення обсягу пакетів завдань для видачі у BBVC-системах.

У цьому випадку всі користувачі-волонтери будуть отримувати пакети завдань однакового розміру під час кожного звернення незалежно від будь-яких обставин:

$$r_{l,k} = r_{l,k-1}, \forall l,k .$$

Дане розбиття є дуже простим в реалізації та надає можливість провести велику кількість оптимізацій та підготовчих процесів до проведення обчислень.

Але ефективність даного методу цілком залежить від правильно підібраної кількості проміжків $r_{l,k}$. При використанні надто великої кількості проміжків ($r_{\max}$), система буде блокувати значну кількість проміжків вже при невеликій кількості волонтерів, що не дозволить залучити нових і негативно вплине на час проведення обчислень. Використання занадто малої кількості проміжків ($r_{\min}$) створить проблему великих витрат на транспортування даних з серверної частини на клієнтську і також збільшить час проведення обчислень. Крім того, є недоцільним надавати однакову кількість проміжків для обчислень волонтерам із різними характеристиками (наприклад, із різними потужностями обчислювальних систем, із різним часом участі у обчисленнях).

Отже, найбільшим недоліком даного методу є відсутність гнучкості (не реагує на події та не враховує характеристики обчислювального процесу користувача), а його ефективність цілком залежить від вибору обсягу пакету завдань.

Способом адаптуватися під особливості кожного користувача-волонтера є використання його найпростішої характеристики – тривалості сесії обчислення.

При ініціюванні користувачем обчислення серверна частина може зберігати час початку сесії обчислення цього користувача та оновлювати його кожен раз при наданні користувачу нового пакету завдань. Це дає можливість обраховувати тривалість сесії обчислення кожного користувача і на її основі визначати обсяг пакету завдань для транспортування відповідному користувачу – чим довшою є сесія користувача, тим більше проміжків йому буде видаватися [2]:

$$r_{l,k} > r_{l,k-1} .$$

Даний метод належить до адаптивних.

Слід зауважити, що починати видачу проміжків варто невеликими кількостями, що відповідає використанню r_{min} із першого методу. З іншого боку, верхнє допустиме значення кількості отриманих проміжків для довготривалих сесій варто обмежити, щоб уникнути блокування великої кількості проміжків одним користувачем, використавши значення $r_{\max}$.

Перевагою даного методу є простота реалізації та хороший приріст швидкодії обчислень для довготривалих сесій користувачів-волонтерів з потужними системами.

Недоліками даного методу є складність обробки виключних ситуацій при роботі з волонтерами та односторонній погляд на тривалість сесії.

Під складністю обробки виключних ситуацій мається на увазі недетермінізм в рішеннях щодо покинутих сесій – в ситуації занадто довгого очікування результатів від користувача системі необхідно вирішувати, чи є дана сесія обчислення покинутою. Сесію можна вважати завершеною та знімати блокування виданих проміжків або через певний час, або якщо періодичні запити до системи користувача завершуються помилкою.

Однобокість погляду на тривалість сесії у даному випадку полягає в розгляді тривалості обчислення тільки як ефективності. Насправді у випадку слабкої системи користувача-волонтера тривалість сесії буде збільшуватись навіть для невеликої кількості опрацьованих проміжків.

Таким чином, розглянуті підходи до визначення обсягу пакету завдань не враховують особливості поведінки волонтера та продуктивність його системи.

3 МАТЕРІАЛИ І МЕТОДИ

Щоб визначити обсяг пакету завдань для кожного волонтера, пропонується врахувати особливості протікання обчислювальних процесів в його системі.

Найпростішим шляхом отримання певних статистичних даних про користувача є аналіз даних про його сесію. Під сесію обчислення вважається процес обчислення завдання з початку створення з'єднання з серверною частиною по проведенню обчислень та закриттям даного з'єднання через будь-які причини (закриття вкладки браузера, перебої в зв'язку).

Під час існування сесії серверна частина може ідентифікувати користувача стандартними засобами бібліотек роботи зі з'єднаннями.

Для ідентифікованого таким чином волонтера l пропонується в якості метрик m, що характеризують його поведінку на i-му зверненні, використати обсяг пакету завдань та час обрахунку кожного проміжку із цього пакету. Зауважимо, що показник $r_{l,i}$ обраховується сервером, а значення $t_{l,i}^{j}, j = \overline{1, r_{l,i}}$ надаються клієнтом при поверненні результатів обчислень.

На основі даних про кожне звернення сервер може порахувати сесійні метрики, в якості яких пропону-

DOI 10.15588/1607-3274-2020-2-16

ється використати кількість обчислених проміжків та тривалість їх обчислення за попередні звернення:

$$R_l(k-1)=\sum_{i=0}^{k-1} r_{l,i}\ ,\ T_l(k-1)=\sum_{i=0}^{k-1} t_{l,i},$$

де $t_{l,i}=\sum_{j=1}^{r_{l,i}} t_{l,i}^{j}$.

Варто зазначити такі властивості поведінкових та сесійних метрик:

– не потребують створення додаткових запитів між клієнтом і сервером для їх передачі;

– можуть зберігатися в структурах даних в пам'яті серверу, що обробляє запити;

– збираються і зберігаються сервером на час сесії користувача і знищуються по її завершенні.

Для оцінки обчислювального процесу введено характеристики $\varepsilon_l(k-1)$ та $a_l(k-1)$. Коефіцієнт $\varepsilon_l(k-1)$ відображає наскільки швидко відбувається обчислення на системі волонтера протягом сесії:

$$\varepsilon_l(k-1)=R_l(k-1)/T_l(k-1).$$

Коефіцієнт $a_l(k-1)$ показує характер зміни швидкості обчислень між зверненнями *k*–2 та *k*–1:

$$a_l(k-1)=\varepsilon_l(k-1)/\varepsilon_l(k-2).$$

Якщо $a_l(k-1)>1$ – обчислення пришвидшуються, $a_l(k-1)<1$ – отримання результатів сповільнюється.

Після того, як визначено набір характеристик обчислювального процесу, для обчислення обсягу пакету завдань пропонується використовувати наступну формулу:

$$r_{l,k}=r_0\cdot(1+\varepsilon_l(k-1))\cdot a_l(k-1), \qquad (1)$$

де $r_0=r_{\min}$.

Таким чином метод обчислення обсягу пакету завдань на основі сесійних метрик користувача полягає у тому, що при кожному зверненні *k* волонтера *l* виконуються кроки 1–4:

1) збираються поведінкові метрики для попереднього звернення $r_{l,k-1}$ та $t_{l,k-1}^{j}$;

2) обчислюються сесійні метрики $R_l(k-1)$ та $T_l(k-1)$;

3) обчислюються характеристики процесу для попереднього звернення $\varepsilon_l(k-1)$ та $a_l(k-1)$;

4) відповідно до формули (1) обчислюється $r_{l,k}$.

Отже, на початку обчислень поведінкові та сесійні метрики, а також характеристики процесу на попередньому кроці відсутні і кожен волонтер отримує мінімальну кількість проміжків. Вже при другому зверненні обсяг даних для кожного волонтера буде змінюватися в залежності від обчислених на основі метрик характеристик обчислювального процесу.

Додатковою перевагою збору поведінкових метрик є те, що на основі аналізу $t_{l,i}$ серверна частина може розпізнати підозрілу активність користувача, ознакою якої є дуже мале значення цього показника. Такий випадок матиме місце, якщо користувач робитиме запити до серверної частини або в обхід браузера, надсилаючи запити на серверну частину вручну, або змінить програмний код клієнтської частини. Тоді варто обмежити максимальну кількість проміжків, які може отримати користувач за один запит. Результуюча формула обрахунку обсягу пакету завдань матиме вигляд:

$$r_{l,k}`=\min(r_{l,k},r_{\max}).$$

Питання визначення мінімальної та максимальної кількості проміжків для транспортування залишається наразі не вирішеним і може стати темою наступних досліджень.

За рахунок зменшення та збільшення обсягу пакету завдань відповідно змінюється і навантаження на систему волонтера. Можна надати користувачу можливість регулювати ступінь використання ресурсів його системи для участі у ВО за рахунок коригування цього обсягу. Тоді результуюча формула матиме такий вигляд:

$$r_{l,k}`=r_{l,k}\cdot k(l),$$

де $r_{l,k}$ отримано в результаті обчислення основною формулою (1), $k(l)\in(0,1]$.

4 ЕКСПЕРИМЕНТИ

Для проведення експериментів було використано існуючу система волонтерських обчислень.

Завдання обчислень полягало у знаходженні хеш-функції над даними, які постійно надходили в систему. Це дало змогу не прив'язуватися до часу виконання обчислень, а зосередитися на характеристиках процесу обчислень. В якості критеріїв оцінки обчислювального процесу використано такі показники:

– сумарна кількість звернень клієнтів до сервера;

– сумарна кількість обчислених клієнтами проміжків.

При виконанні експерименту прийнято:

– $\|d_i\|=1$;

– $r_{\min}=10$;

– $r_{\max}=100$;

– час проведення обчислень – 5 хвилин.

Моделювання проводилося за наступних умов:

– кожне завдання видається для перевірки одному волонтеру;

– перевірка результатів обчислень сервером відсутня.

Тестування алгоритмів планування в реальних обчислювальних системах є складним завданням, тому для цього часто використовують емулятори або іміта-

DOI 10.15588/1607-3274-2020-2-16

ційні моделі [5]. Для моделювання роботи BBVC-системи використано технологію контейнеризації Docker, бібліотеку Puppeteer та браузер Chrome.

Проведено експерименти за участі 10 та 20 волонтерів. Для 50% учасників вхідні дані надавалися методом однакової кількості проміжків, для інших 50% обсяг вхідних даних обчислювався з урахуванням їх сесійних метрик.

Оскільки обрахунок відбувався дуже швидко, клієнти додатково генерували затримку в діапазоні 100–200 мс перед поверненням результатів.

Щоб характеристики процесів обчислень можна було порівняти, розрахунок обсягу даних із застосуванням обох методів виконувалися одночасно. Для спостереження за метриками було використано програмний інструмент Prometheus.

5 РЕЗУЛЬТАТИ

Проведене моделювання підтвердило характер процесів при використанні розглянутих методів планування завдань.

Так, у системі із наданням однакової кількості проміжків приріст сумарної кількості звернень до сервера залежить від кількості клієнтів, а приріст сумарної кількості обчислених проміжків залежить від обраного на початку обчислень обсягу пакету завдань.

Для системи із сесійними метриками використання штучних затримок значно спростило моделювання, а також надало можливість відстежувати зміну обсягів пакетів завдань. Так при значному збільшенні значення штучної затримки обсяги пакетів завдань теж значно зменшувалися, і навпаки, при встановленні значення затримки в початкове, система поверталася до попереднього значення обсягів пакетів.

Приріст сумарної кількості запитів до сервера в такій системі не залежить від кількості клієнтів, тобто вона буде краще масштабуватися. Оскільки потужнішим клієнтам видавався більший обсяг пакету завдань, вони довше проводили обчислення і менше зверталися до сервера.

Приріст сумарної кількість обчислених проміжків в такій системі залежить від потужності систем користувачів: чим потужнішими є системи, тим більший приріст має цей показник. При зниженні потужностей волонтерів приріст цього показника відповідно зменшується.

На рис. 1, 2 показано графіки навантаження на сервер (сумарної кількості запитів до сервера) та кількість обчислених результатів (сумарної кількості обчислених проміжків) для випадку 5+5 волонтерів.

Результати моделювання свідчать про наступне:

– навантаження на сервер за умови надання клієнтам однакової кількості проміжків зростає більш стрімко, ніж з урахуванням метрик;

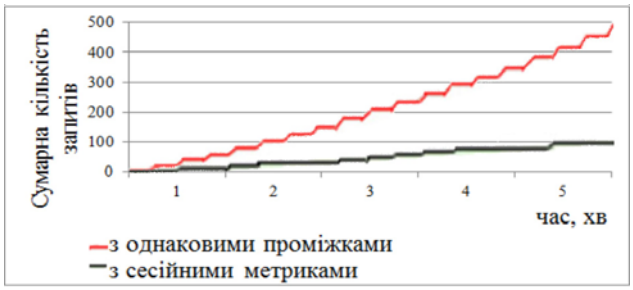

Рисунок 1 – Навантаження на сервер

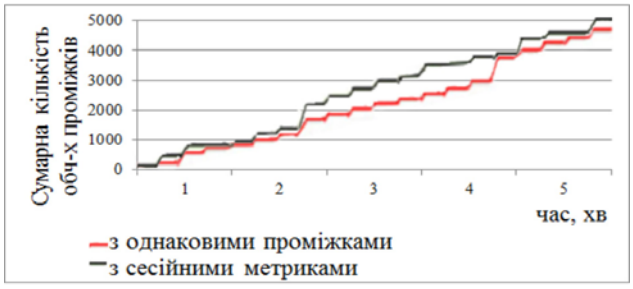

Рисунок 2 – Кількість обчислених результатів

– кількість обчислених результатів при використання метрик є дещо більшою, ніж при наданні однакової кількості проміжків за умови однакової тривалості обчислень.

6 ОБГОВОРЕННЯ

Порівняння методів планування завдань з наданням однакової кількості проміжків та використанням сесійних метрик користувача показало, що запропонований метод дозволяє підвищити ефективність систем ВО:

– значно зменшується навантаження на сервер;

– швидкість отримання результатів обчислень не зменшується.

Проведені експериментальні дослідження підтвердили простоту програмної реалізації та ефективність запропонованого методу. При цьому можна визначити такі його основні переваги над «наївними» методами:

– обсяг пакету завдань підлаштовується під кожного волонтера;

– реагує на зміну характеристик обчислювального процесу;

– зменшується вплив початкового розміру пакету завдань на характеристики процесу обчислень.

Також вказаний метод має переваги над методами з використанням інформації про вузли та збором статистики:

– не потребує спеціальних сховищ для зберігання метрик;

– не потребує додаткової інформації про дані та ресурси.

Реалізація можливості волонтера впливати на ступінь завантаженості своєї системи через корегування обсягу пакету завдань додає гнучкості розробленій програмній системі.

Дана праця є продовженням та розвитком досліджень [3].

DOI 10.15588/1607-3274-2020-2-16

ВИСНОВКИ

Вирішується проблема підвищення ефективності обчислень у BBVC-системах за рахунок корегування обсягу пакету завдань відповідно до характеристик обчислювального процесу кожного користувача – учасника обчислень.

Наукова новизна отриманих результатів полягає в тому, що вперше запропоновано адаптивний метод планування завдань, який базується на використанні сесійних метрик. Його особливістю є те, що обсяг пакетів завдань підлаштовується під кожного користувача, враховуючи характеристики обчислювального процесу на його системі. Запропонований метод дозволяє підвищити ефективність проведення обчислень за рахунок зменшення навантаження на сервер та зменшення кількості блокованих проміжків вхідних даних.

Практичне значення отриманих результатів полягає в тому, що формулу обчислення обсягу пакету завдань на основі метрик для було інтегровано в існуючу систему для проведення розподілених волонтерських обчислень, що надало можливість зменшити кількість мережевих запитів при одночасному збільшенні кількості обчислених результатів.

Перспективи подальших досліджень полягають у використанні статистичних даних користувача та інформації від його браузеру для обчислення обсягу пакету завдань.

ЛІТЕРАТУРИ/ ЛИТЕРАТУРЫ

1. Pan Y. Gray Computing: A Framework for Distributed Computing with Web Browsers: thesis doctor of philosophy / Pan Yao. – Vanderbilt University, 2017. – 152 p. URL: http://etd.library. vanderbilt.edu/available/etd-11192017-220210/
2. https://www.zdnet.com/article/could-smartphones-replace-datacenters-these-finnish-researchers-think-so/
3. Рибачок Н. А. Моделювання поведінки систем організації волонтерських обчислень у браузері з використанням WebWorkers / Н. А. Рибачок // Управляючие системы и машины. – 2019. – № 1 (279). – С. 76–87. DOI: https://doi.org/10.15407/usim.2019.01.076
4. Lightweight Volunteer Computing Platform using Web Workers / [P. Chorazyk, M. Godzik, K. Pietak et al.] // ICCS 2017: International Conference on Computational Science, Zürich, 12–14 June 2017: proceedings. – 2017. – P. 948–957. DOI: https://doi.org/10.1016/j.procs.2017.05.091
5. Чернов И. А. Обзор методов планирования заданий в Desktop Grid / И. А. Чернов, Е. Е. Ивашко, Н. Н. Никитина // Программные системы: теория и приложения. – 2017. – № 1. – С. 3–29. DOI: https://doi.org/10.25209/2079-3316-2017-8-3-3-29
6. Task Scheduling in Desktop Grids: Open Problems / I. Chernov, N. Nikitina and E. Ivashka // Open Engineering. – 2017. – Vol. 7, Issue 1. DOI: https://doi.org/10.1515/eng-2017-0038
7. Computational models and heuristic methods for grid scheduling problems / F. Xhafa, A. Abraham // Future Generation Computer Systems. – Vol. 26, Issue 4. – 2010. –P. 608–621. DOI: https://doi.org/10.1016/j.future.2009.11.005
8. Debski R. ComcuteJS: A Web Browser Based Platform for Large-scale Computations / R. Debski, T. Krupa, and P. Majewski // Computer Science (AGH). – 2013. – Vol. 14(1). – P. 143–152. DOI: http://dx.doi.org/10.7494/csci.2013.14.1.143
9. Hadoop. http://hadoop.apache.org/
10. Pando: Personal Volunteer Computing in Browsers. / [E. Lavoie, L. Hendren, F. Desprez et al.] // Middleware'19: 20th ACM/IFIP International Middleware Conference 2006, Davis, 9–13 December 2019: proceedings. – 2019. – P. 96–109. URL: https://arxiv.org/abs/1803.08426
11. Fabisiak T. Browser-based Harnessing of Voluntary Computational Power / T. Fabisiak, A. Danilecki // Foundations of Computing and Decision Sciences. – 2017. – Vol. 42, Issue 1. – P. 3–42. DOI: https://doi.org/10.1515/fcds-2017-0001
12. Estrada T. "Challenges in designing scheduling policies in volunteer computing" in Desktop Grid Computing / T. Estrada, M. Taufer, C. Cerin, G. Fedak. – CRC Press, 2012. – P. 167–190.

Received 28.02.2020.
Accepted 20.04.2020.

УДК 004.75

ИСПОЛЬЗОВАНИЕ СЕССИОННЫХ МЕТРИК ПРИ ПЛАНИРОВАНИИ ЗАДАЧ В СИСТЕМАХ ВОЛОНТЕРСКИХ ВЫЧИСЛЕНИЙ В БРАУЗЕРЕ

Рыбачок Н. А. – канд. техн. наук, старший преподаватель кафедры программного обеспечения компьютерных систем Национального технического университета Украины «Киевский политехнический институт имени Игоря Сикорского», Киев, Украина.

Орос Б. Б. – магистрант кафедры программного обеспечения компьютерных систем Национального технического университета Украины «Киевский политехнический институт имени Игоря Сикорского», Киев, Украина.

АННОТАЦИЯ

Актуальность. Исследование посвящено разработке адаптивного метода планирования задач в системах распределенных волонтерских вычислений в браузере. Это дает возможность выдавать каждому волонтеру пакет задач для вычислений в зависимости от характеристик его вычислительного процесса. При этом более производительные клиенты будут получать больший объем задач, болем эффективно используя предоставленные ресурсы, уменьшая нагрузку на сервер и способствуя масштабированию вычислений; менее производительные клиенты будут получать меньший объем задач, решая проблему избыточных блокировок входных данных.

Цель. Целью работы является повышение эффективности систем распределенных волонтерских вычислений в браузере путем использования сессионных метрик при планировании задач.

Метод. В работе описаны сессионные метрики, которые определяют особенности поведения пользователей-волонтеров. На их основе введены показатели, характеризующие ход вычислительного процесса на системах пользователей. Предложена формула для вычисления объема пакета заданий для распределения клиенту, которая учитывает производительность системы клиента и тенденции протекания процессов в его системе. На основании существующей системы волонтерских вычислений выполнено моделирование вычислений с использованием различных методов расчета объема пакетов задач.

DOI 10.15588/1607-3274-2020-2-16

Для оценки эффективности таких систем использовано общее количество обращений пользователей к серверной части и общее количество вычисленных результатов.

Результаты. Предложенный подход применен в существующей системе проведения волонтерский вычислений. Сравнение результатов работы предложено метода с существующими показало уменьшение нагрузки на сервер и ускорения получения результатов вычислений при использовании сессионных метрик.

Выводы. Проведенное моделирование подтвердило удобство и эффективность использования сессионных метрик при планировании задач. Представленный подход обеспечивает возможность масштабирования систем распределенных волонтерских вычислений. Перспективы дальнейших исследований заключаются в использовании статистических данных пользователя и информации от его браузера для вычисления объемов данных передаваемых клиенту.

КЛЮЧЕВЫЕ СЛОВА: распределенные волонтерские вычисления в браузере, планирование задач, пакет задач, распределение задач, сессионные метрики.

UDC 004.75

USAGE OF SESSION METRICS TO TASKS SCHEDULING IN BROWSER-BASED VOLUNTEER COMPUTING SYSTEMS

Rybachok N. A. – PhD, Senior Lecturer, Computer Systems Software Department, National Technical University of Ukraine "Igor Sikorsky Kyiv Politechnic Institute", Kyiv, Ukraine.

Oros B. B. – Master student of Computer Systems Software Department, National Technical University of Ukraine "Igor Sikorsky Kyiv Politechnic Institute", Kyiv, Ukraine.

ABSTRACT

Context. The study is devoted to the development an adaptive method for scheduling tasks in distributed browser-based volunteer computing systems. which is constructed on the session metrics of a volunteer user. This allows server to allocate tasks to each volunteer for computing, depending on the characteristics of his computing processes. More productive clients will receive more tasks, maximizing resource utilization, reducing server load and facilitating scaling; less productive clients will receive fewer tasks, solving the problem of excessive blocking of input data.

Objective. The aim of the study is increasing the efficiency of distributed browser-based volunteer computing systems by using session metrics to tasks scheduling.

Method. The session metrics that determine the behavior of volunteer users are described.

The indicators that characterize the progress of the computational process on user systems were introduced. The formula to calculate the amount of input data for distribution to client is proposed. It takes into account the capacity of the client's system and the tendency of processes on its system. Simulation was done using different methods to calculate the amount of input data on the basis of the existing system of volunteer computing. To evaluate the performance of such systems, the total number of user hits to the server part and the total number of calculated results were used.

Results. The proposed method is applied in the existing browser-based volunteer computing system. A comparison of the results of the proposed method with the existing ones showed a decrease in server load and an increase of the number of calculated results when session metrics are used.

Conclusions. The simulation confirmed the convenience and efficiency of session metrics usage to tasks scheduling. The presented approach provides scalability of distributed browser-based volunteer computing systems. The prospects of further research are to use the users' statistics and information from their browsers to calculate the amount of input data for distribution to client.

KEYWORDS: distributed browser-based volunteer computing, tasks scheduling, tasks package, tasks distribution, session metrics.

REFERENCES

1. Pan Y. Gray Computing: A Framework for Distributed Computing with Web Browsers: thesis doctor of philosophy. Vanderbilt University, 2017, 152 p. URL: http://etd.library. vanderbilt.edu/available/etd-11192017-220210/
2. https://www.zdnet.com/article/could-smartphones-replace-datacenters-these-finnish-researchers-think-so/
3. Rybachok N. A. Modelyuvannya povedinky system orhanizatsiyi volonters′kykh obchyslen′ u brauzeri z vykorystannyam WebWorkers, *Upravlyayuchye systemy y mashyny*, 2019, No. 1 (279), pp. 76–87. DOI: https://doi.org/10.15407/usim.2019.01.076
4. Chorazyk P., Godzik M., Pietak K. et al. Lightweight Volunteer Computing Platform using Web Workers, *ICCS 2017: International Conference on Computational Science, Zürich, 12–14 June 2017: proceedings*, 2017, pp. 948–957. DOI: https://doi.org/10.1016/j.procs.2017.05.091
5. Chernov Y. A., Yvashko E. E., Nykytyna N. N. Obzor metodov planyrovanyya zadanyy v Desktop Grid, *Prohram-mnye systemy: teoryya y prylozhenyya,* 2017, No. 1, pp. 3–29. DOI: https://doi.org/10.25209/2079-3316-2017-8-3-3-29
6. Task Scheduling in Desktop Grids: Open Problems / I. Chernov, N. Nikitina and E. Ivashka // Open Engineering. – Vol. 7, Issue 1. – 2017. https://doi.org/10.1515/eng-2017-0038
7. Xhafa F., Abraham A. Computational models and heuristic methods for grid scheduling problems / // *Future Generation Computer Systems,* 2010, Vol. 26, Issue 4, pp. 608–621. https://doi.org/10.1016/j.future.2009.11.005
8. Debski R., Krupa T., and Majewski P. ComcuteJS: A Web Browser Based Platform for Large-scale Computations, *Computer Science (AGH),* 2013, Vol. 14(1), pp. 143–152. http://dx.doi.org/10.7494/csci.2013.14.1.143
9. Hadoop. http://hadoop.apache.org/
10. Lavoie E., Hendren L., Desprez F. et al. Pando: Personal Volunteer Computing in Browsers, *Middleware"19: 20th ACM/IFIP International Middleware Conference 2006, Davis, 9–13 December 2019: proceedings*, 2019, pp. 96–109. URL: https://arxiv.org/abs/1803.08426
11. Fabisiak T., Danilecki A. Browser-based Harnessing of Voluntary Computational Power, *Foundations of Computing and Decision Sciences*, 2017, Vol. 42, Issue 1, pp. 3–42. DOI: https://doi.org/10.1515/fcds-2017-0001
12. Estrada T., Taufer M., Cerin C., Fedak G. Challenges in designing scheduling policies in volunteer computing in Desktop Grid Computing, CRC Press, 2012, pp. 167–190.

DOI 10.15588/1607-3274-2020-2-16