УДК 004.272.43

О.Н. РОМАНЮК, В. В. ВОЙТКО, С. С. ВОЗНИЙ, О.О. МОРФІЯНЕЦЬ
Вінницький національний технічний університет

ВИКОРИСТАННЯ ГРАФІЧНИХ ПРОЦЕСОРІВ ДЛЯ ОБРОБКИ СЕМАНТИЧНИХ ДАНИХ

Анотація – у статті розглянуто можливості використання відеокарт для неграфічних розрахунків, таких як швидке опрацювання символів у системі ASCII, мультипроцесорний перебір, паралельне порівняння слів, сортування тексту за ознаками. Наведені приклади побудови аналізатора на основі графічних процесорів. Продемонстровано можливість обробки значних об'ємів даних у паралельному режимі.

Ключові слова: CUDA, семантика, символи, текст, переклад, мультипроцесори.

OLEKSANDER ROMANYK, VIKTORIYA VOYTKO, SERHII VOZNYI, OLEG MORFIYANEC
Vinnytsia National Technical University

USING GRAPHIC PROCESSOR FOR PROCESSING OF SEMANTIC DATA

The article deals with the possibility of using graphics cards for non-graphical calculations, such as fast processing of the symbols in the system ASCII, multiprocessor bust, parallel comparison of words, sorting of the text attributes. Imposed examples of the construction of the analyzer based on GPUs. Demonstrated the possibility processing large amounts of data in parallel.

Keywords - CUDA, semantics, symbols, text, translation multiprocessors.

Вступ. Загальна постановка проблеми

Сучасний розвиток Інтернет-технологій обумовлює поширення електронного документообігу, орієнтованого на активне використання веб-ресурсів. Всесвітня мережа поділяється на доменні зони, які мають прив'язку за певними територіальними та мовними ознаками. Міжнародне співробітництво і бізнес світ потребують ведення документації трьома мовами: мовами сторін, що укладають договори, контракти, угоди, та англійською як інтернаціональною мовою. Тому питання якісного автоматизованого перекладу є актуальним і перспективним, враховуючи тенденцію попиту на послухи перекладачів і лінгвістів [1].

У семантичних обрахунках значну роль відіграють мовні корпуси, що акумулюють інформацію про особливості конкретної мови. Використання корпусів дає змогу з певною достовірністю здійснювати переклад текстів.

На сучасному етапі розвитку існує велика кількість різноманітних програмних та апаратних рішень семантичного аналізу та перекладу текстових ресурсів. Проте переважна більшість з них дозволяє працювати з обмеженими об'ємами інформації, не підтримує автоматизований переклад та використовується для перекладів окремих документів з відомою мовою тексту.

Актуальність теми зумовлена потребою комерційних організацій знаходити рішення, які б дозволяли обробляти великі масиви інформації, коли обробки потребують сотні Гб текстових даних. Такий технологічний процес перекладу потребує чимало ресурсів. Для аналізу семантичних даних з невизначеною мовою текстів обумовлюється використання серверних систем великої потужності. Використання семантичних корпусів великих об'ємів тягне за собою значні фінансові затрати та потребу постійного моніторингу стану системи [2]. Саме за таких умов використання потужності відеокарт забезпечує збільшення продуктивності обрахунків та зменшення затрат часу на аналіз тексту та його переклад.

Метою роботи є підвищення ефективності обробки великих масивів даних шляхом використання відеокарт у семантичних операціях, орієнтованих на багатопоточне паралельне порівняння. Головними задачами вважаємо дослідження можливих варіантів побудови семантичних аналізаторів на основі GPU, порівняння складності побудови та принципів роботи семантичних аналізаторів, реалізованих на серверних системах, та семантичних аналізаторів на основі GPU, дослідження можливості оптимізації процесів семантичного аналізу, що в перспективі дасть змогу створювати ефективні системи машинного перекладу великих об'ємів даних.

Огляд методів символьного аналізу

Методи семантичного аналізу використовують мовні корпуси або спеціалізовані бази даних, які різняться мовною, змістовною, культурною, тематичною приналежністю [3]. Найпоширенішим методом ідентифікації мови є метод повного перебору за словником, орієнтований на циклічний перебір усіх слів тексту з організацією їх пошуку в мовній базі даних. За кількістю виявлених збігів визначається мова тексту. Цей метод забезпечує високу достовірність результату, проте його основним недоліком є потреба у використанні великих об'ємів пам'яті та ресурсів мікропроцесора [1,2].

Метод символьного аналізу [1] не потребує великих затрат пам'яті. У своїй роботі він оперує кодами символів у системі ASCII. Підчас аналізу тексту окремо розглядається кожен символ, а не ціле слово, що знімає потребу використання мовних корпусів. Недоліками методу є тривалий час обробки символів та потреба в паралельних обчисленнях [3].

Корпусна лінгвістика - розділ мовознавства, що займається розробкою, створенням та використанням текстових корпусів. Лінгвістичним корпусом називають сукупність текстів, зібраних за

певними принципами, розмічених за стандартом і забезпечених спеціалізованою пошуковою системою [3].

Використання корпусу є ключовим у процесі перекладу тексту. Проте мовна ідентифікація не потребує використання цілого корпусу, вона опирається на словники, враховує символьні особливості конкретних мов, що забезпечуються символьними таблицями ASCII та таблицями спеціальних символів й ієрогліфів. Виявлення характерного коду символів свідчить про належність до певної мовної групи. Такий метод має обмежене використання через велику кількість можливих помилок у процесі мовної ідентифікації, відсутність засобів визначення діалектичних ознак мовної приналежності. Цю задачу вирішує метод повного словникового перебору.

Незалежно від обраного методу процес визначення мови тексту потребує використання багатопроцесорних комп'ютерів для ведення обрахунків. Сам процес мовної ідентифікації є досить трудомістким, тому на процесорах з відносно малою кількістю ядр (менше 4) є не достатньо ефективним [2,3].

Необхідність встановлення мови методом перебору потребує застосування потужних багатопроцесорних систем. Вирішити цю задачу можна за рахунок використання відеокарти персонального комп'ютера.

Визначення різниці між CPU і GPU в паралельних розрахунках

Зростання частоти універсальних процесорів має фізичне обмеження і характеризується високим енергоспоживанням, тож збільшення продуктивності процесорів усе частіше відбувається за рахунок розміщення декількох ядер в одному чіпі. Сучасні процесори містять до чотирьох ядер і використовують MIMD - множинний потік команд і даних. Кожне ядро працює окремо, виконуючи різні інструкції для різних процесів.

Поява в універсальних процесорах спеціалізованих векторних можливостей (SSE2 і SSE3) для чотирикомпонентних (з одинарною точністю обчислень з плаваючою крапкою) і двокомпонентних (з подвійною точністю) векторів обумовлена стрімким ростом вимог до графічних додатків.

У відеочіпах NVIDIA основним блоком постає мультипроцесор з вісьмома-десятьма ядрами, сотнями ALU, декількома тисячами регістрів і невеликим об'ємом поділюваної загальної пам'яті. Сама відеокарта містить швидку глобальну пам'ять з доступом до неї всіх мультипроцесорів, локальну пам'ять кожного окремого мультипроцесора та спеціальну пам'ять для збереження констант [4,5].

Окремо відзначимо, що наявні ядра мультипроцесора в GPU є SIMD ядрами з одиночним потоком команд і безліччю потоків даних. Ці ядра виконують одні й ті ж інструкції одночасно. Такий стиль програмування є звичайним для реалізації графічних алгоритмів та вирішення широкого кола наукових задач, але вимагає впровадження спеціалізованих прийомів програмування. В той же час такий підхід дозволяє збільшити кількість виконавчих блоків за рахунок спрощення їх структури. Універсальні процесори оптимізовані для досягнення високої продуктивності виконання єдиного потоку команд, обробляють різноформатні дані (цілі числа і числа з плаваючою крапкою) та забезпечують випадковий доступ до пам'яті [5].

Розробники CPU намагаються забезпечити паралельне виконання якомога більшої кількості інструкцій з метою збільшення продуктивності роботи процесорів. Так, починаючи з процесорів Intel Pentium, з'явилося суперскалярне виконання, що забезпечує реалізацію двох інструкцій за такт, а Pentium Pro відзначився позачерговим виконанням інструкцій. Проте паралельне виконання послідовного потоку інструкцій має певні базові обмеження, тож прямим збільшенням кількості виконавчих блоків кратного збільшення швидкості не домогтися [1,4].

Класично GPU отримує на вході групу полігонів, проводить операції їх базової обробки і на виході видає пікселі. Безпосередня обробка полігонів і пікселів є незалежною, тож їх обробляють паралельно, окремо один від одного. Тому паралельна організація роботи GPU орієнтована на використання великої кількості виконавчих блоків, які легко завантажити, на відміну від послідовного потоку інструкцій для CPU. Крім того, сучасні GPU забезпечені можливістю виконання більш, ніж однієї інструкції за такт (подвійний випуск). Так, архітектура Tesla за деяких умов запускає на виконання операції MAD + MUL або MAD + SFU одночасно.

Реалізація семантичного аналізу тексту на GPU

Сьогодні перспективним є напрям використання відеокарт для виконання операцій семантичного порівняння та сортування, що дозволяє за малий проміжок часу проводити порівняння значних масивів даних, сортувати їх за обраними ознаками та проводити паралельні обчислення [2,4-7].

Так використання GPU у семантичному аналізі дозволяє паралельне порівняння в ALU символів з тексту. При цьому слід окремо виділити процесори для підрахунку виявлених відповідностей кодів символів з еталонними символьними корпусами. В цьому випадку можна буде проводити паралельне порівняння цілих масивів символів і відслідковувати ідентифіковані збіги. По завершенню процесу інформаційно-семантичного аналізу порівнюються індекси виявлених збігів за корпусами. Найбільший індекс ідентифікує належність тексту до конкретної мови.

При використанні GPU у семантичних багатопоточних розрахунках забезпечується можливість реалізації декількох режимів роботи аналізатора, що полягає в опрацюванні потоку даних багатьма логічними процесорами, при чому до кожного з ALU буде застосовуватися однакова командна інструкція. Аналізатор такого типу можна побудувати на основі SIMD (Single Instruction, Multiple Data), що забезпечує

одиночний потік команд та множини потік даних. Це елемент класифікації за таксономією Флінна для паралельних процесорів [5,9], де до багатьох елементів даних застосовується одна чи однакові команди. Таким чином, SIMD є однією з головних умов, що гарантують можливість паралельного виконання алгоритмів (рис 1).

Інформаційним ресурсом аналізатора постає лінгвістичний символьний або мовний корпус. До кожного ALU буде застосована процедура порівняння слів з тексту із повним перебором слів з корпусу. Також окремо слід виділити процесор для підрахунку індексів співпадань та процесор для визначення результатів порівняння індексів. Аналізатор орієнтований на виконання порівняльних процесів повного словникового перебору слів з тексту зі словами мовного корпусу. Структуру аналізатора зображено на рис. 2.

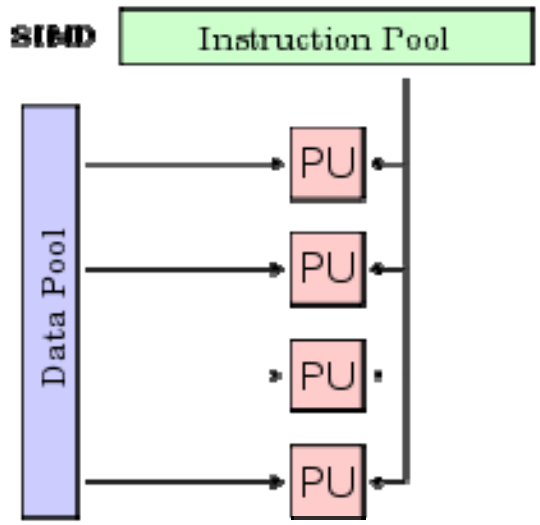

Рис. 1. Структура SIMD

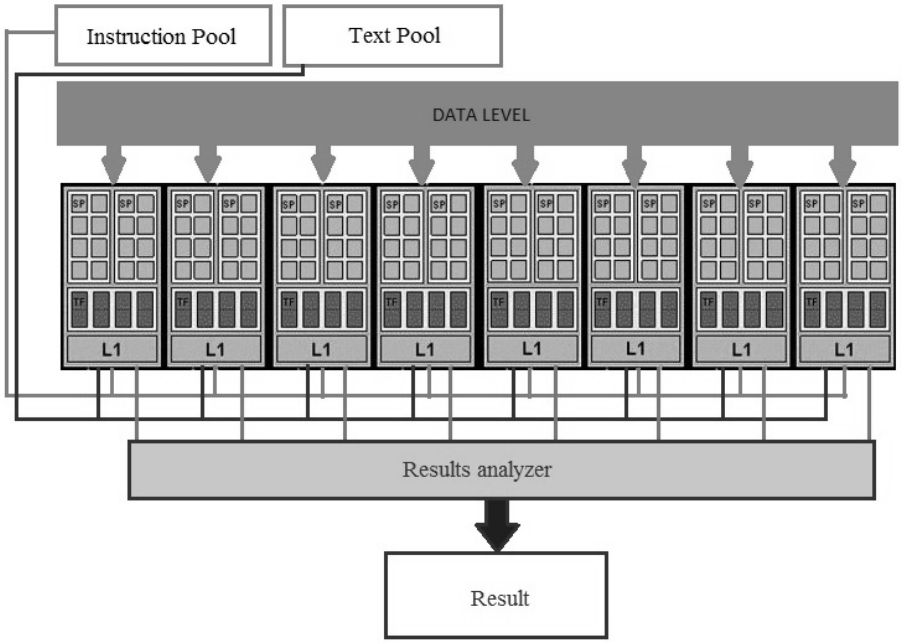

Рис. 2. Структура семантичного аналізатора на основі GPU

Побудований аналізатор визначає мову тексту за методом повного семантичного порівняння значно швидше, ніж аналоги, реалізовані на CPU. Крім того, слід очікувати підвищення результатів достовірності мовної ідентифікації. Недоліком такого аналізатора є повний доступ до інформаційного рівня всіх ALU. У такому випадку слід очікувати, що різні ALU будуть проводити аналіз однакових слів та показувати однакові індекси співпадань. Вирішити цю проблему можна, розділивши інформаційний рівень кожного ALU на частини. При такому розподіленні можлива поява помилок у роботі ALU у випадку, коли ALU не зможе знайти співпадання слів з даними інформаційного рівня. Щоб добитися коректного аналізу у таких ситуаціях, слід зробити інформаційні блоки для кожного ALU рівносильними. Цього можна досягнути, використавши генератор випадкових чисел і змішавши мовні одиниці таким чином, щоб усі частини Data level були рівнозначними. При такому режимі роботи слід вважати достовірним той результат, який матиме хоча б 10% співпадань з інформаційним рівнем. Розподілений аналізатор описаного типу вважає операцію мовної ідентифікації завершеною тоді, коли хоча б на одному аналізаторі індекс співпадань перевищить 1/10 частину усього тексту, що аналізується. Умова завершення операції аналізу для змішаного джерела даних описується виразом

$$10\% \geq \frac{t}{T} \times 100\% ,$$

де t – проаналізований текст на одному із ALU, T – загальний обсяг тексту, що аналізується.

Ефективна робота аналізатора зі змішаним джерелом даних для порівняння досягається за умови виконання порівняльних операцій з типовим мовним корпусом кількістю ALU, меншою за 10. В іншому випадку можливе виникнення повторного аналізу однакових частин тексту на різних ALU. Це може призвести до виникнення петель та тупикових ситуацій, коли умова завершення не буде виконуватися

взагалі. Для уникнення такої ситуації слід контрольовано виділяти під процес обробки однієї частини текстових даних кількість ALU, меншу за 10. Схема побудови семантичного аналізатора описаного типу подана на рис.3.

Через обмежене використання ALU в роботі такого аналізатора не виключена можливість появи певної кількості вільних ALU. Це створює проблему нераціонального використання апаратних ресурсів відеокарт. Означена проблема обумовлює необхідність відслідковування завантаженості потоків обробки даних.

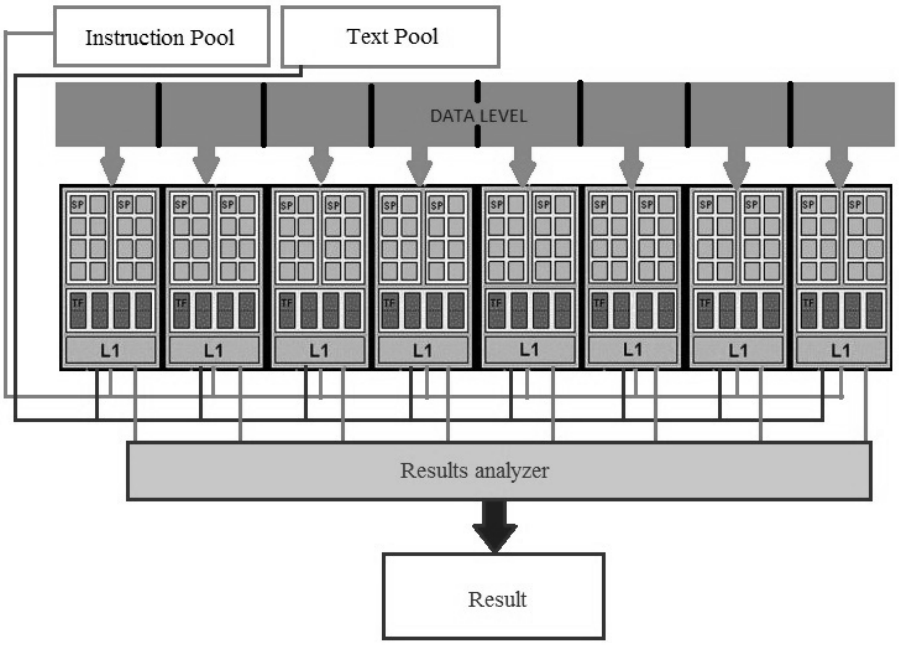

Рис. 3. Структура семантичного аналізатора на основі GPU із розподіленням інформаційного рівня між ALU

Задача швидкої мовної ідентифікації масиву документів, що складається з мов різних регіонів, потребує проведення одночасного аналізу даних за всіма належними лінгвістичними корпусами. У випадку використання CPU для семантичного аналізатора знадобилося використання окремого серверного комп'ютера для кожної лінгвістичної групи. Для найпоширеніших лінгвістичних груп (англомовної, західноєвропейської, східноєвропейської, діалектів китайського) потрібно чотири серверних машини. При використані GPU лінгвістичний аналізатор можна реалізувати на основі MISD-архітектури (Multiple Instruction stream, Single Data stream) - архітектури паралельних обчислень, де кілька функціональних модулів (два чи більше) виконують різні операції над одними даними (рис. 4). Для аналізаторів такого типу під інформаційним рівнем розуміємо текстовий файл, що аналізується [7,8,9].

Робота аналізатора буде полягати в тому, щоб порівняти повний обсяг аналізованого масиву даних з усіма наявними мовними корпусами одночасно. Процес аналізу вважається успішно завершеним у випадку виконання умови (1):

$$\begin{cases} \frac{t}{T} = 1; \\ K > \left(\sum_{i=1}^{n} k_i\right) - K \end{cases} \quad (1)$$

де t – проаналізований текст, T – вхідний текст для аналізу, n – кількість ALU, використана п процесі аналізу, K – коефіцієнт співпадань на конкретному ALU, k_i – коефіцієнти співпадань на ALU в момент оцінки стану [6,7,9].

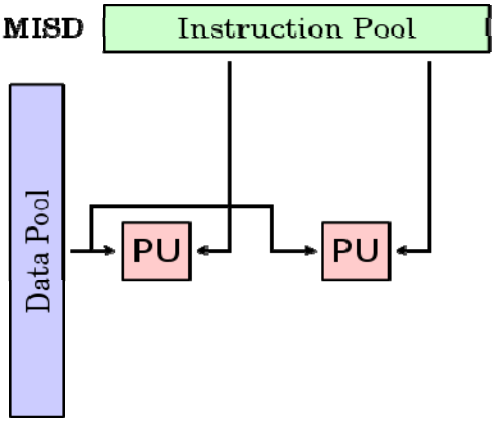

Рис. 4. Архітектура MISD

Кожна одиниця інформації з робочого файлу одночасно проходить аналіз в усіх наявних лінгвістичних корпусах, задіяних в процесі порівняння даних. Можливу реалізацію паралельного аналізатора наведено на рис. 5.

Висновки

Теоретичний аналіз та практичні випробування показали, що використання GPU для семантичних обрахунків дозволяє значно пришвидшити процес аналізу текстових даних, відкриває можливості для подальших досліджень у галузі машинного перекладу, пошуку нестандартних підходів до вирішення проблем паралельної обробки великих масивів даних.

У роботі запропоновано можливі реалізації семантичних аналізаторів на основі GPU. Розроблено варіанти функціювання аналізаторів залежно від потреб мовної ідентифікації. Проаналізовані можливості обробки значних масивів текстових даних за допомогою відеокарти.

Розробка семантичних аналізаторів на основі GPU забезпечить швидке та більш точне визначення мови текстів.

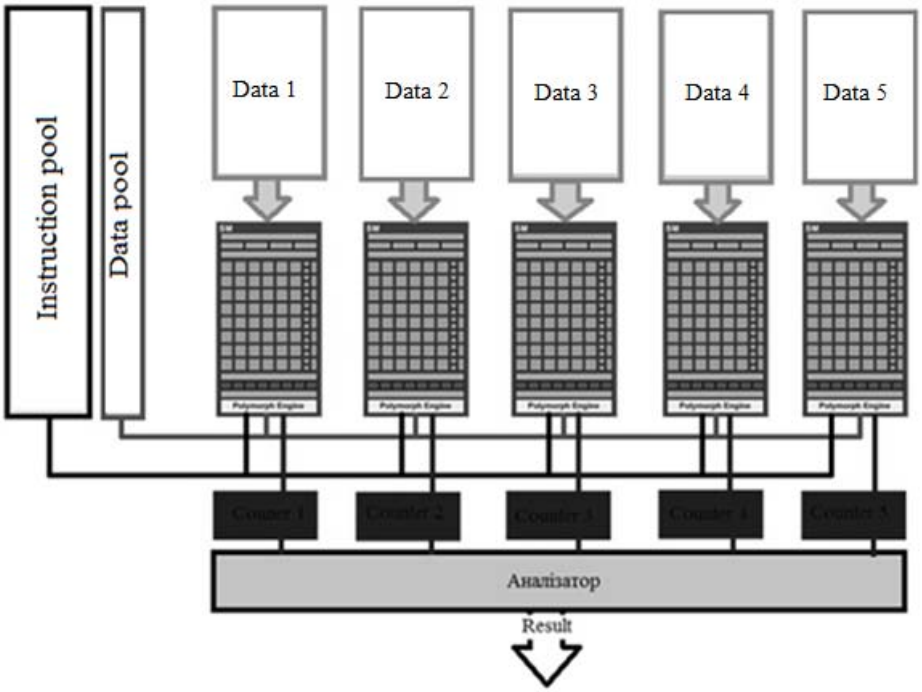

Рис. 5. Структура паралельного аналізатора

Література

1. Dovnar PY, Alexander Vorontsov Linguistic processor. Development / P. Yu Dovnar / / International Journal of Computer Science with: information systems and technology: Proceedings of the International Scientific Congress. - 2011. - Vol. 15. - S. 18-96.

2. Straka AM Corpus Linguistics (using the hull definition text) [E resource] / A. Straka / / Corpus Linguistics. Access mode: http://ru.wikipedia.org/wiki/ASCII.

3. Mitin OV, AS Evdokimenko Methods of text analysis: methodological foundations and program implementation [E resource] / O. Mitin, AS Evdokimenko. - Mode of access: http://cyberleninka.ru/article/n/metody-analiza-teksta-metodologicheskie-osnovaniya-i-programmnaya-realizatsiya. - Title screen.

4. History of machine translation [E resource] / / History of machine translation. Access mode: http://mrtranslate.ru/guessers.php.

5. Jennings T. An annotated history of some character codes or ASCII American Standard Code for Information Infiltration / T. Jennings, / / American Standard Code for Information Infiltration. - 2004. - № 1. - Pp. 11 ¬ -66.

6. National versions ASCII [E resource] / / National versions ASCII. Access mode: http://ru.wikipedia.org/wiki/ASCII.

7. Smirnov Non-graphic computing on the graphics card (NVIDIA CUDA and AMD Stream) [E resource] / M. Smirnov // Non-graphic computing on the graphics card. Mode of access: http://poisk-videokart.ru/article/articles/negraficheskie-vychisleniya-na-videokarte-nvidia-cuda-i-amd-stream/19.html

8. Berillo A. NVIDIA CUDA - non-graphic computing on GPU [E resource] / A. Berillo / / NVIDIA CUDA - non-graphic computing on GPUs. Access mode: http://www.ixbt.com/video3/cuda-1.shtml

9. CUDA.NET for. NET-developer [E resource] / / CUDA.NET for. NET-developer. Access mode: http://habrahabr.ru/sandbox/22562/.

References

1. Dovnar PY, Alexander Vorontsov Linguistic processor. Development / P. Yu Dovnar / / International Journal of Computer Science with: information systems and technology: Proceedings of the International Scientific Congress. - 2011. - Vol. 15. - S. 18-96.

2. Straka AM Corpus Linguistics (using the hull definition text) [E resource] / A. Straka / / Corpus Linguistics. Access mode: http://ru.wikipedia.org/wiki/ASCII.

3. Mitin OV, AS Evdokimenko Methods of text analysis: methodological foundations and program implementation [E resource] / O. Mitin, AS Evdokimenko. - Mode of access: http://cyberleninka.ru/article/n/metody-analiza-teksta-metodologicheskie-osnovaniya-i-programmnaya-realizatsiya. - Title screen.

4. History of machine translation [E resource] / / History of machine translation. Access mode: http://mrtranslate.ru/guessers.php.

5. Jennings T. An annotated history of some character codes or ASCII American Standard Code for Information Infiltration / T. Jennings, / / American Standard Code for Information Infiltration. - 2004. - № 1. - Pp. 11 ¬ -66.

6. National versions ASCII [E resource] / / National versions ASCII. Access mode: http://ru.wikipedia.org/wiki/ASCII.

7. Smirnov Non-graphic computing on the graphics card (NVIDIA CUDA and AMD Stream) [E resource] / M. Smirnov / / Non-graphic computing on the graphics card. Mode of access: http://poisk-videokart.ru/article/articles/negraficheskie-vychisleniya-na-videokarte-nvidia-cuda-i-amd-stream/19.html

8. Berillo A. NVIDIA CUDA - non-graphic computing on GPU [E resource] / A. Berillo / / NVIDIA CUDA - non-graphic computing on GPUs. Access mode: http://www.ixbt.com/video3/cuda-1.shtml

9. CUDA.NET for. NET-developer [E resource] / / CUDA.NET for. NET-developer. Access mode: http://habrahabr.ru/sandbox/22562/.

Рецензія/Peer review : 26.9.2014 р. Надрукована/Printed : 16.11.2014 р.