УДК 004.89

О.В. Олещук, О.Є. Попель, М.Б. Копитчук
Одеський національний політехнічний університет

МОДЕЛЮВАННЯ НЕЙРОННИХ МЕРЕЖ КОХОНЕНА НА ГРАФІЧНОМУ ПРОЦЕСОРІ

Розглядається інтеграція таких інформаційних технологій, як нейронні мережі і паралельні обчислення на графічному процесорному ядрі з використанням технології CUDA. Коротко описано нейронної мережі Кохонена і карт, що самоорганізуються. Запропонована модель цієї мережі, що враховує особливості апаратної платформи і дає змогу реалізувати нейронну мережу Кохонена на графічному процесорному ядрі, отримавши за рахунок цього істотний приріст продуктивності.

Ключові слова: нейронні мережі Кохонена; карти, що самоорганізуються; GPU загального призначення; технологія CUDA.

This article discusses the integration of information technology such as neural networks and parallel computing on the graphic processor core using the CUDA technology. A brief description of the neural network and Kohonen self-organizing maps is adduced. A model of the network, which takes into account characteristics of the hardware platform, and allows realizing of the Kohonen neural network in the graphic processor core, due to this received substantial performance gains.

Key words: neural networks, fully connected neural network, general purpose GPU, CUDA technology.

Постановка проблеми

Нейромережеві технології є потужним математичним апаратом, що уможливлює виявляти приховані закономірності. Подібні задачі необхідно розв’язувати у багатьох предметних областях, таких як медицина, геологічна розвідка, аналіз складних економічних систем тощо. При цьому, якщо в окремих областях припустимо проводити обробку початкових даних у відкладеному режимі, то у багатьох інших, наприклад, при діагностиці актуального стану технічної системи, необхідно проводити аналіз у режимі реального часу або у режимі, близькому до нього.

До теперішнього часу застосування нейромережевих технологій під час роботи з великими масивами даних багато у чому обмежувалося лабораторними дослідженнями у режимі відкладеного часу, оскільки програмна реалізація моделі нейронної мережі відрізнялася низькою швидкістю роботи або було потрібно розробити спеціалізовані апаратні засоби. Тому ставиться завдання – опанувати типовими сучасними апаратними засобами і повною мірою задіяти їхні можливості для забезпечення істотного приросту продуктивності нейронної мережі.

Аналіз літературних джерел

Одним з найперспективніших апаратних засобів, на які можна покласти обчислювальне навантаження з моделювання нейронної мережі, є графічне процесорне ядро (GPU).

Спочатку відеокарта з вбудованими у неї обчислювальними ресурсами була призначена винятково для перетворення цифрових даних, що обробляються центральним процесором, в аналоговий сигнал для монітора. Але у процесі розвитку обчислювальних систем і підвищення складності та обсягу графічної інформації, відеоадаптер еволюціонував у порівняно незалежний високопродуктивний спеціалізований обчислювач. Характерними його особливостями є [4]:

- високий паралелізм;
- система команд SIMD;

- швидке перемикання потоків;
- велика кількість порівняно простих ALU;
- первинна орієнтація на дійсні числа одинарної точності.

За рахунок цих властивостей багато завдань, які в природний спосіб можуть бути розбиті на незалежні або порівняно незалежні підзадачі, можуть розв’язуватися з використанням GPU на порядки швидше, ніж з використанням потужностей центрального процесора. У той самий час зазначені особливості GPU вимагають певного підходу до програмування графічного ядра для забезпечення його якнайповнішого завантаження.

Для управління графічним процесором були розроблені спеціалізовані програмні засоби, такі як DirectX і OpenGL, що уможливлювали використовувати GPU як пристрій, який ефективно виконує графічні операції. Такі програмні засоби вже можна було використовувати і для неграфічних обчислень, але це було пов'язано з низкою незручностей, наприклад, з необхідністю кодувати числові дані у графічному вигляді і виконувати зворотне перетворення. Тому з часом були розроблені такі технології, як OpenCL і CUDA, що надають доступ програмісту до обчислювальних ресурсів графічного ядра і при цьому уможливлюють обробляти безпосередньо числові масиви. Отже, можна розглядати технологію CUDA як інформаційну технологію, яка використовує обчислювальні ресурси графічного процесорного ядра для істотного підвищення продуктивності обчислювальної системи загалом.

Розробляючи програми для GPU, необхідно враховувати архітектурні особливості апаратури і правильно структурувати дані, розподіляючи їх між глобальною та спільною пам'яттю [2, 3], і враховувати обмеження, що стосуються можливості одночасного доступу до комірок цих видів пам'яті.

Мета роботи – ознайомитись з основними особливостями і принципами роботи нейронних мереж Кохонена і карт, що самоорганізуються, з подальшим розглядом їх реалізації на графічному процесорі з використанням програмних можливостей, що надаються технологією CUDA.

Загальні відомості про нейронні мережі Кохонена

Нейронні мережі Кохонена – це широкий клас нейронних мереж, що містить у собі так званий шар Кохонена [1]. Шар Кохонена являє собою множину формальних нейронів з лінійною функцією активації (рис. 1).

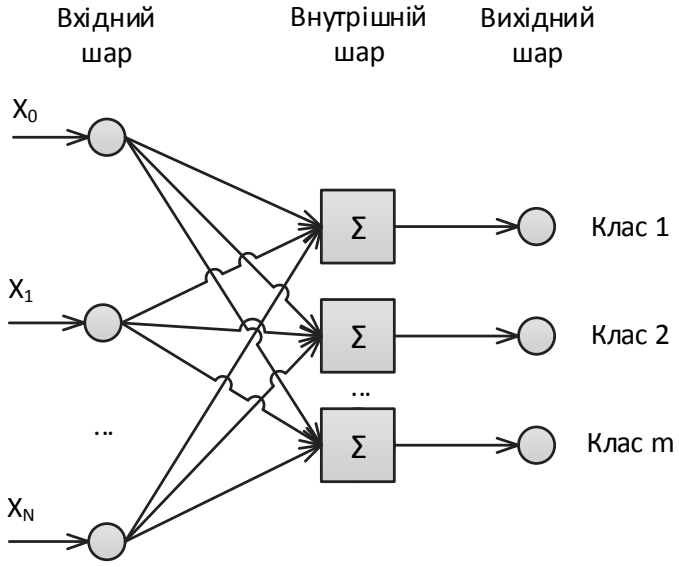

Рис. 1. Загальний вигляд нейронної мережі Кохонена

Нейрони вхідного і вихідного шарів не виконують жодних математичних обчислень, а просто є точками, з яких можна отримати і розподілити значення сигналу. Прихований шар обчислює вихідне значення за формулою

$$y_k = \sum_{i=1}^{N} w_{ik} \cdot x_i ,$$

де y_k – вихід k-го нейрона; w_{ik} – вага синапса між i-м входом та k-м нейроном; x_i – значення i-го входу; N – кількість вхідних нейронів.

Характерною особливістю мереж Кохонена є принцип "переможець забирає все", тобто максимальне вихідне значення дорівнює одиниці, а усі інші – нулю.

Важливим різновидом мереж Кохонена є мережі, що самоорганізуються. На додаток до зв'язків, наявних у класичній мережі Кохонена, показаній на рис. 1, мережа, що самоорганізується, має структуру зв'язків між вихідними нейронами, задану симетричною таблицею мір сусідства. Вага кожної пари нейронів з номерами i та j представлена величиною η_{ij}, де $0 \le \eta_{ij} \le 1$. Максимальна вага відповідає найближчому елементу, а мінімальні ваги сусідства будуть у нейронів, рознесених по протилежних кутах карти.

Розмірність карти, що самоорганізується, може бути довільною, але найпопулярнішими є двовимірні карти, особливо під час роботи з даними, що мають прив'язку до географічного положення, наприклад, під час геологічної розвідки, під час діагностики територіально-розподілених, але механічно пов'язаних систем, таких як нафто- і газопроводи.

За рахунок використання принципу сусідства нейрони у картах, що самоорганізуються, шикуються не хаотично, як у класичній мережі Кохонена, а розташовуються так, що нейрони, пов'язані з класами з близькими значеннями параметрів, знаходяться поряд. А це дає змогу згодом виконати над отриманою картою ручний або автоматичний кластерний аналіз. З цієї ж причини навіть у задачах, не пов'язаних з геометрією або картографією, часто є сенс використовувати двовимірні карти, що самоорганізуються, оскільки їх зручно графічно показати на екрані і тим самим отримати наочне уявлення про кластери.

Одним з основних показників продуктивності нейронної мережі є швидкість її навчання, оскільки для більшості їх різновидів навчання має ітеративний характер і вимагає багаторазового пред'явлення зразків з навчальної вибірки. Для нейронних мереж Кохонена цей показник ще важливіший, тому основною областю їх застосування є попередній аналіз даних з метою їх класифікації та виявлення прихованих закономірностей. На відміну від класичних нейронних мереж прямого поширення, таких як персептрон Розенблатта, результат навчання нейронної мережі Кохонена, представлений у зручному, найчастіше графічному вигляді, вже дає змогу досліднику отримати відповідь на його питання. З цієї причини навчання мережі Кохонена часто виявляється єдиною ресурсомісткою обчислювальною операцією.

Існує багато підходів до підвищення швидкості і якості навчання нейронних мереж Кохонена, наприклад, завдання початкових вагових коефіцієнтів, залежно від значень у початковій вибірці, зашумлення вхідних векторів, зміна числа корегованих нейронів, наділення нейронів "почуттям справедливості" [1]. Усі ці методи ґрунтуються виключно на математичній моделі мережі Кохонена (рис. 1), яка визначає її функціональність, але не враховує архітектуру обчислювальних засобів, на якій вона буде реалізована.

У роботі пропонується інший підхід – використовувати природний паралелізм нейронних мереж, що особливо яскраво проявляється у нейронній мережі Кохонена, і реалізовувати їх на апаратній базі, що має такі самі властивості, а саме: на основі графічних процесорів загального призначення. Такий підхід аніскільки не суперечить відомим принципам і уможливлює одночасно з ним використовувати існуючі методи підвищення продуктивності нейронних мереж.

Реалізація нейронної мережі Кохонена з використанням технології CUDA

Процес навчання нейронної мережі Кохонена можна подати у вигляді трьох етапів: обчислення відстані між поданим на мережу зразком і кожним нейроном мережі (рис. 2), визначення переможця і зміни вагових коефіцієнтів. У картах Кохонена, що самоорганізуються, присутній також додатковий етап – визначення мір сусідства.

Для знаходження відстані може бути використана довільна метрика. Якщо розв'язувана задача має географічну прив'язку або якщо здійснюється лише попереднє ознайомлення з предметною областю і її властивості не з'ясовані, то часто використовують евклідову метрику. Тоді відстань визначається за формулою

$$d_k = \sqrt{\sum_{i=1}^{N} (w_{ik} - x_i)^2} \ .$$

З метою підвищення швидкості навчання операцію вилучення квадратного кореня можна виключити, оскільки у кінцевому підсумку інтерес являє не власне значення відстані, а співвідношення відстаней.

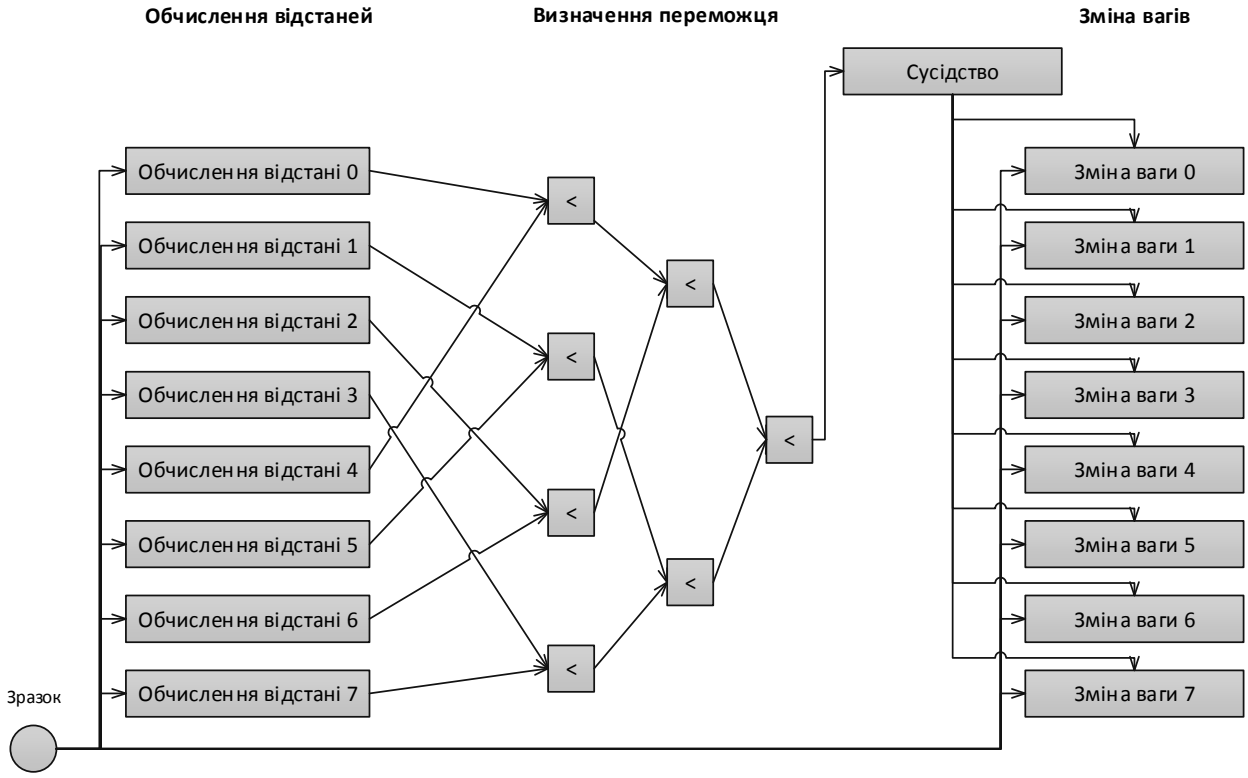

Рис. 2. Модель карти Кохонена, що самоорганізується, реалізованої з використанням технології CUDA

Модель на рис. 2 показана з низкою припущень. Тут, як приклад, показана мережа, що складається з восьми нейронів, але загальна її структура за зміни кількості нейронів істотно не зміниться. Передбачається, що кількість вхідних нейронів не задана умовами задачі і може бути обрана довільно. У цьому випадку її доцільно задавати, враховуючи оптимальну кількість одночасно виконуваних потоків на графічному ядрі. Використання меншої кількості нейронів призведе до того, що графічне ядро буде завантажене не повністю, і отже, не буде досягнута максимально можлива продуктивність обчислювальної системи. Збільшення числа нейронів реалізується доволі просто за умови, що їх число є кратним кількості одночасно працюючих потоків. У цьому випадку кожен елемент буде прораховувати два або більше нейронів послідовно. Не рекомендується використовувати кількість нейронів, не кратну кількісті потоків, оскільки, крім простоювання частини процесорних елементів, це вимагатиме додавання умовних операторів, що у технології CUDA є небажаним, з огляду на продуктивність.

Наступний етап – знаходження переможця. Для цього необхідно знайти найменшу відстань і визначити індекс відповідного йому нейрона. Цей етап може бути представлений у вигляді задачі сходження, тобто обчислення такого роду, коли на кожному кроці кількість оброблюваних даних зменшується вдвічі. Найкращим способом розбиття цього етапу на складові буде поділ масиву вихідних даних навпіл і попарне порівняння між собою елементів, розташованих послідовно у кожній половині масиву. Саме такий вибір елементів для порівняння пов'язаний з особливостями організації глобальної та спільної пам'яті [3].

У разі роботи з класичною мережею Кохонена, виявивши переможця, можна буде сказати, у якого саме з нейронів вихідне значення повинно дорівнювати одиниці, а значить, можна відразу приступати до зміни вагових коефіцієнтів за формулою

$$w_{ik} = w_{ik} + \lambda(x_i - w_{ik}),$$

де λ – коефіцієнт швидкості навчання.

Цей процес з обчислювальної точки зору виконується аналогічно до етапу визначення відстаней, тобто кожен елемент прораховує вагу одного нейрона, і ці розрахунки виконуються паралельно.

Послідовність з цих трьох етапів виконується багаторазово. На кожній ітерації на вхід подається новий зразок, який, як правило, обирається довільно. Також загальноприйнятою практикою є поступове зменшення коефіцієнта швидкості навчання.

Трохи окремо стоїть процес визначення мір сусідства, який використовується у картах, що самоорганізуються. Його особливість полягає у тому, що виконати його можна одноразово перед початком навчання мережі, оскільки сусідство між нейронами не залежить від їх вагових коефіцієнтів, а визначається виключно, враховуючи індекси нейронів і вибраного типу координатної сітки – прямокутної або гексагональної. За порівняно малої кількості нейронів попереднє обчислення заходів сусідства цілком доцільне, тому таблицю сусідства можна зберігати у загальній пам'яті. За збільшення кількості нейронів обсягу загальної пам'яті може виявитися недостатньо. Задіяти для таблиці сусідства глобальну пам'ять немає сенсу, оскільки доступ до неї є одним з “вузьких” місць в архітектурі CUDA. Тому в останньому випадку продуктивнішим рішенням буде виконання розрахунку сусідства між кожним з нейронів і нейроном-переможцем заново на кожній ітерації.

Аналіз продуктивності

Для тестування були використані карти Кохонена, що самоорганізуються, розміром 16x16, 16x32 і 32x32. Вони містили три вхідні сигнали і один вихідний. У цьому випадку смислове наповнення сигналів значення немає. Тобто це можуть бути сигнали від датчиків температури, тиску, вібрації тощо. Закони розподілу значень у сигналах, частота їх змін, взаємна кореляція та інші показники безпосередньо на швидкість навчання мережі не впливають, однак вони можуть надати непрямий вплив, оскільки за меншої зашумленості і за менш складного закону взаємозв'язку вхідних сигналів для виявлення прихованих закономірностей достатньо буде меншої кількості ітерацій під час навчання.

Вихідним сигналом у мережі Кохонена є певна цільова функція, наприклад, кількість палива, споживаного двигуном, або швидкість руху транспортного засобу. На швидкісні характеристики мережі особливості вихідного сигналу також надають тільки опосередкований вплив, за рахунок різного ступеня кореляції з вхідними сигналами, що відповідно позначиться на кількості ітерацій, необхідних для навчання.

На рис. 3 показана карта, що самоорганізується, на різних стадіях навчання. Початковий її стан (рис. 3, *а*) являє собою повністю зашумлену область, оскільки початкові ваги нейронів, як правило, призначаються випадково. Протягом навчання карти на ній поступово вимальовуються великі темні і світлі плями, оскільки нейрони з близькими вагами прагнуть наблизитися один до одного (рис. 3, *б*), а потім межі областей стають чіткіше вираженими, і часто при цьому можна спостерігати кілька кластерів (рис. 3, *в*). За продовження навчання кластери стають все дрібнішими аж до окремих нейронів (рис. 3, *г*). Такий стан, як правило, свідчить про певне перенавчання мережі. Враховуючи представлені рисунки, можна зробити висновок, що за вибраних числових характеристик карти, що самоорганізується, зокрема, за обраного розміру, необхідна кількість ітерацій її навчання знаходиться приблизно у діапазоні 200 тис. – 1 млн. ітерацій.

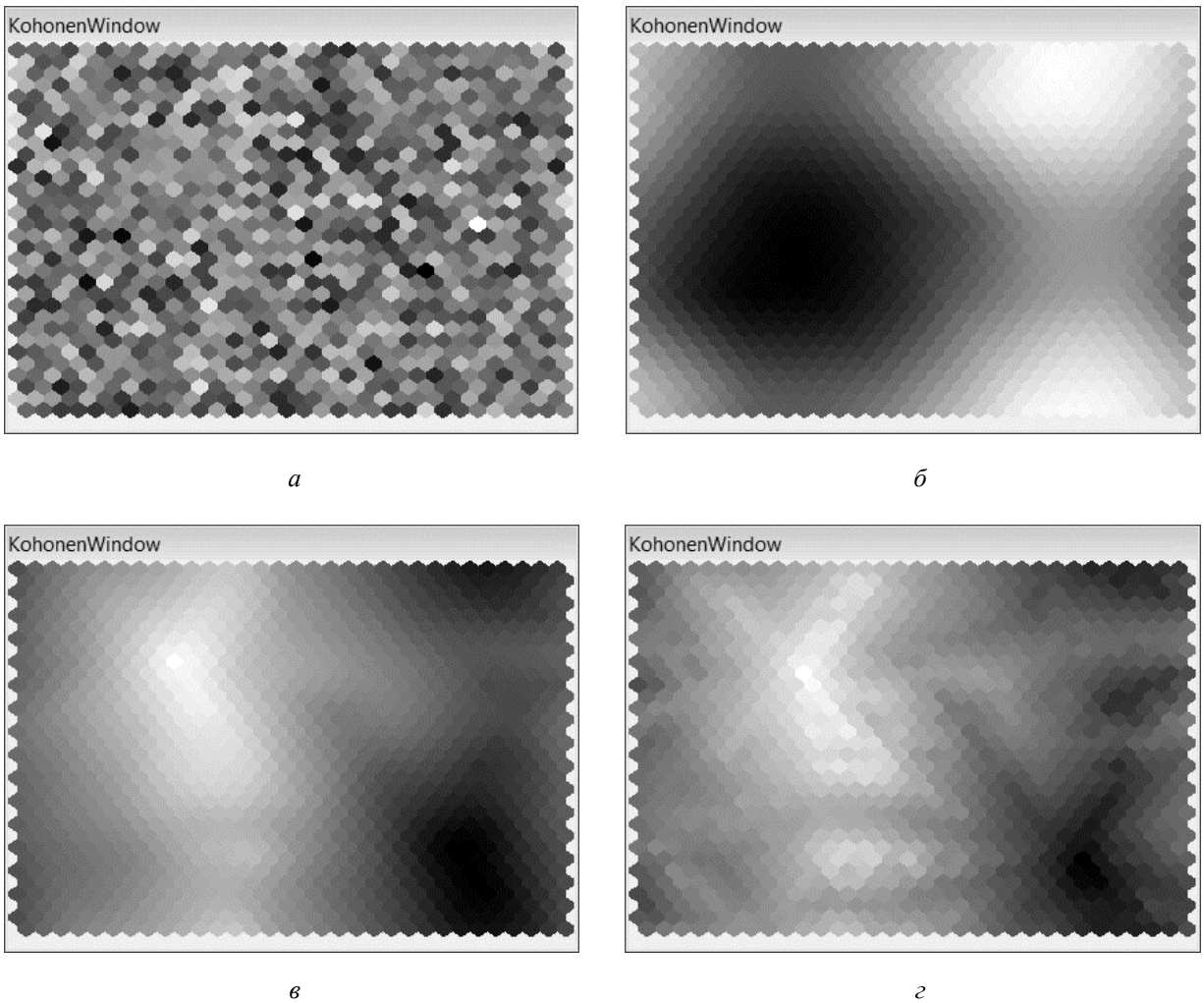

Рис. 3. Етапи навчання карти Кохонена, що самоорганізується:
а – початковий стан; б – 10 тис. ітерацій; в – 200 тис. ітерацій; г – 1 млн. ітерацій

Результат тестування зображено на рис. 4. У цьому випадку фіксувалася кількість навчальних ітерацій, а саме – встановлювалася рівною 1 млн. і оцінювався час, необхідний для їх виконання. Як бачимо, в усіх випадках час не перевищив 100 с. Подібна кількість ітерацій у багатьох випадках буде більшою, ніж потрібно для попереднього налаштування мережі, а отриманий час навчання – цілком прийнятним для інтерактивної роботи оператора з обчислювальною системою.

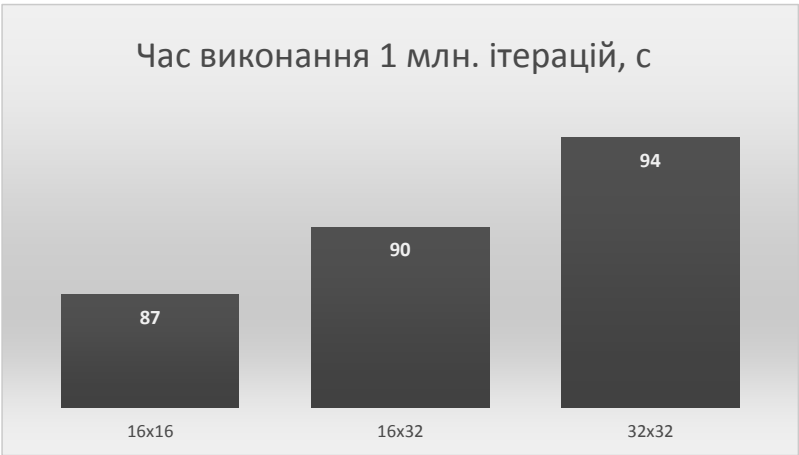

Рис. 4. Продуктивність карти Кохонена, що самоорганізується,
реалізована з використанням технології CUDA

Також бачимо, що за зменшення розміру карти, яка самоорганізується, час навчання знижується незначно. Наприклад, за переходу від карти розміром 32x32 до карти розміром 16x16, тобто за зменшення кількості нейронів у 4 рази, час навчання знизився приблизно на 7 %. Така поведінка нейронної мережі, реалізованої відповідно до моделі на рис. 2, цілком узгоджується з передбаченою, оскільки одночасно зі зниженням кількості нейронів зменшувалася і кількість працюючих потоків у GPU.

Розмір навчальної вибірки у проведеному тестуванні дорівнював 65536. Ключовим тут є те, що такий обсяг даних вже не може бути розміщений у загальній пам'яті, отже, передбачається, що масив зразків навчальної вибірки розміщений у глобальній пам'яті відеокарти. Враховуючи числові характеристики сучасних відеокарт і середніх обсягів вибірок у більшості задач, цю ситуацію можна вважати найхарактернішою. За збільшення обсягу навчальної вибірки може здійснюватися довантаження даних з ОЗУ (за необхідності) у пам'ять відеокарти безпосередньо у процесі обчислень на GPU. Істотного впливу на продуктивність це не матиме.

Висновок

На етапі розробки моделі карти, що самоорганізується, і за вибору її кількісних показників, таких як кількість нейронів і кількість елементів у навчальній вибірці, має бути враховано багато обмежень і особливостей GPU. У цьому випадку реалізація карти Кохонена, що самоорганізується, з використанням графічного процесора на основі технології CUDA дає змогу досягти швидкості навчання мережі близько 10^5 ітерацій за секунду, що відкриває перспективи використання карт, що самоорганізуються, у системах м'якого реального часу.

1. Кохонен Т. Самоорганизующиеся карты // пер. с англ В. Агеев; под ред. Ю. Тюменцева. – М.: Бином, 2008. – 656 с. 2. Гергель В.П. Теория и практика параллельных вычислений. – М.: Бином. Лаборатория знаний, 2007. – 424 с. 3. Олещук О.В., Попель О.Є., Копитчук М.Б. Моделювання повнозв'язної нейронної мережі з використанням технології CUDA // Вісник Національного університету "Львівська політехніка". – 2012. – № 747. – С. 131–139. 4. CUDA Zone. http://www.nvidia.ru/object/cuda_home_new_ru.html (Last access: 05.10.2013).