МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ЗАПОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

Факультет математики

Кафедра комп'ютерних наук

КВАЛІФІКАЦІЙНА РОБОТА МАГІСТРА

на тему: "Розробка програмного забезпечення для прогнозування та аналізу часових рядів"

Виконав: студент 2 курсу групи 8.1221
Спеціальності: 122 комп'ютерні науки
Освітньої програми: комп'ютерні науки
С.Ю Кабанов
Керівник: професор кафедри
комп'ютерних наук, д. т. н.
Шило Г.М

Запоріжжя 2022

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ЗАПОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

Факультет математичний

Кафедра Комп'ютерних наук

Рівень вищої освіти магістр

Спеціальність 122 комп'ютерні науки

(шифр і назва)

Освітня програма комп'ютерні науки

ЗАТВЕРДЖУЮ

Завідувач кафедри загальної математики, к.ф.-м.н., доцент

_____ Чопоров С.

(підпис)

« ____ » _____ 2022 р.

З А В Д А Н Н Я

НА КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТОВІ(СТУДЕНТЦІ)

Кабанову Семену Юрійовичу

(прізвище, ім'я та по-батькові)

1. Тема роботи (проекту) Розробка програмного забезпечення для прогнозування та аналізу часових рядів

керівник роботи (проекту) Шило Галина Миколаївна

(прізвище, ім'я та по-батькові, науковий ступінь, вчене звання)

затверджені наказом ЗНУ від «____» _____ 2022 року № _____

2. Строк подання студентом роботи 2022

3. Вихідні дані до роботи 1. Постановка задачі.

2. Перелік літератури.

3.Теоретична частина

4.Розробка

5.Висновок

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)

Аналіз розробки алгоритму

Розробка алгоритму

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)

Презентація

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв

7. Дата видачі завдання 12.10.2022

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1.	Розробка плану роботи.	15.10.2022	
2.	Збір вихідних даних.	18.10.2022	
3.	Обробка методичних та теоретичних	20.10.2022	
	джерел.		
4.	Розробка першого розділу.	26.10.2022	
5.	Розробка другого розділу.	01.11.2022	
6.	Оформлення та нормоконтроль		
	кваліфікаційної роботи.	05.11.2022	
7.	Захист кваліфікаційної роботи.	13.12.2022	

Студент _____ С.Ю.Кабанов
(підпис) (ініціали та прізвище)

Керівник роботи _____ Шило Г.М
(підпис) (ініціали та прізвище)

Нормоконтроль пройдено

Нормоконтролер _____ О.Г. Спиця
(підпис) (ініціали та прізвище)

РЕФЕРАТ

Кваліфікаційна робота магістра «Розробка програмного забезпечення для прогнозування та аналізу часових рядів»: 63 с., 10 рис., 7 табл., 33 джерел, 1 додатки.

Розробка програмного забезпечення для прогнозування та аналізу часових рядів

Об’єкт дослідження –Часові ряди та нейронні мережі.

Мета роботи: розробка програми прийняття рішень.

Метод дослідження – аналітичний, порівняльний.

У цій роботі буде розглянуто методи та алгоритми системи прийняття рішень і розробка програми для цієї системи.

SUMMARY

Master’s Qualification Thesis «Тема роботи згідно наказу»: 63 pages, 10 figures, 6 tables, 33 references, 1 supplements.

Development of software for forecasting and analysis of time series

The object of the study is Time series and neural networks.

The aim of the study is development of a decision-making program.

The research method is analytical and comparative.

This work will consider the methods and algorithms of the decision-making system and the development of the program for this system.

ЗМІСТ

ВСТУП

Зі збільшенням обсягів інформації та збільшенням складності технічної, соціальної, економічної та інших систем стає все важче людині, яка бере участь в управлінні цими системами, оцінювати усю вхідну в органи управління і безпосередньо перед відповідальним за прийняття рішень, інформацію, а також оцінювати вплив багатьох різних факторів на поведінку системи.

Для досягнення найкращих результатів, як за якістю, так і в кількісному плані, мінімізуючи при цьому ризики серйозних втрат і нещасних випадків, в сучасній науці активно розвивається напрямок систем підтримки прийняття рішень.

Передумови для використання систем підтримки прийняття рішень найчастіше є такими факторами:

- жорстка конкуренція, немає права на помилку;
- необхідність швидкого планування
- необхідність швидкого виявлення негативних тенденцій;
- необхідно проаналізувати в режимі реального часу великий обсяг інформації для прогнозування та оцінки поточного стану керованої або спостережуваної системи.

Одним з найпоширеніших завдань використання систем підтримки прийняття рішень є прогнозування.

Слід зазначити, що більшість систем знаходяться в постійному розвитку - їх подальший стан залежить від ряду попередніх держав і від послідовності їх проходження. Системи можуть бачити тенденції періодичних змін і циклічності.

Незважаючи на те, що було розроблено велику кількість систем підтримки прийняття рішень та статистичних застосувань для прогнозування часового ряду, рідкісні здатні виробляти значущі результати у великих системах зі складними взаємозалежностями та недовизначеними даними. Підвищення точності та швидкості прогнозування та підвищення якості результатів,

створення більш швидкої та точнішої системи підтримки прийняття рішень завжди залишатиметься актуальним, поки решта гравців не відреагують на ситуацію.

Метою цієї роботи є розробка оболонки (без прив'язки до певного домену) системи підтримки прийняття рішень для прогнозування значень часового ряд, який міг би працювати з недовизначенними даними.

Можливість роботи з невизначеними даними в даній оболонці досягається за допомогою роботи з інтервативами.

Підвищення точності для систем зі складними з'єднаннями реалізується за рахунок використання зв'язку з інтервальної нейронної мережею і генетичних алгоритмів. У роботі розглядається використання нейронних мереж і генетичних алгоритмів окремо, і пропонується принципово нова схема їх взаємодії.

Продуктивність швидкості покращується за допомогою паралельної обчислювальної архітектури на процесорах графічних адаптерів (відеокартах). Для цього було проаналізовано архітектуру паралельних обчислень та розроблено метод паралельної обробки отриманого пучка еволюційних алгоритмів.

Наукова новизна полягає в тому, щоб підвищити точність параметрів керованої або спостережуваної системи, необхідній їй, а також збільшити норму розрахунків, необхідних для прогнозування або оцінки.

Практична важливість роботи полягає в розробці програмного продукту, який реалізує отримані методи.

1 ЗАГАЛЬНІ ВІДОМОСТІ ПРО ОБЛАСТЬ ТЕМИ

1.1. Аналіз тематики системи підтримки прийняття рішень

Ювену, який приймає рішення, часто має справу з системою складних залежностей між різними факторами або ресурсами, які впливають на їх цілі або бажані результати в процесі пошуку найкращого рішення і прогнозування можливих результатів. Для того, щоб прийняти правильне рішення, яке найкращим чином відповідає поставленим умовам і поставленим цілям, спостережувана система повинна бути піддана якісному комплексному аналізу. Чим глибше відповідальний за прийняття рішень вивчить всі фактори і врахує їх вплив один на одного, тим кращим буде рішення або передбачення. Чим складнішою є система і чим більше факторів, що сприяють, тим складніше і трудомісткіше її вивчати.

Складна ***система*** відноситься до системи, яка взаємодіє з частинами (підсистемами), які, в свою чергу, також є системами підсистем. то і аналіз системи буде складним.

Прийняття рішень щодо складних систем (таких як управління організованості, промислові процеси, інвестиційні портфелі внутрішні або контроль атомних електростанцій тощо) часто є занадто складним для когнітивних здібностей людини.

Існує багато емпіричних доказів того, що людське інтуїтивне судження і рішення, яке вона приймає, може бути далеким від оптимального. Кількість помилок і неточностей, зроблених людиною, збільшується пропорційно складності системи і обернено пропорційна кількості часу, відведеного на прийняття рішення. Оскільки в багатьох випадках якість і швидкість прийняття рішень є життєво важливими, допомога при прийнятті рішень стала одним з основних застосувань науки і техніки в останні десятиліття. Наукові дисципліни, такі як статистика, економіка та дослідження транзакцій, розробили достатню

кількість різних методів, спрямованих на вибір раціонального рішення. Останнім часом якість цих методів була покращена за допомогою накопичених знань в таких сферах, як інформатика, когнітивна психологія і штучний інтелект. Запропоновані методи і алгоритми реалізуються у вигляді комп'ютерних програм, як автономні інструменти або як інтегровані обчислювальні середовища для комплексного рішення. Загальна назва таких середовищ – системи підтримки прийняття рішень (SPRs). Найбільш загальне визначення SPPR дано в літературі. ***SPPR*** **-** це інтерактивна комп'ютерна система, яка допомагає користувачам судити про процес, вибирати рішення або прогнозувати результати.

Системи підтримки прийняття рішень набирають популярності в різних галузях, включаючи бізнес, інженерію, військових і медицину. Вони особливо цінні в ситуаціях, коли обсяг наявної інформації значно перевищує спроможність людині і в завданнях, для яких точність і швидкість мають значення. Системи підтримки прийняття рішень можуть допомогти людині вивчити предметну область, інтегруючи різні джерела інформації, надаючи доступ до необхідних інтелектуальних знань, допомагаючи структурувати рішення. Вони також можуть підтримувати вибір серед чітко визначених альтернатив, спираючись на формальні підходи, такі як інженерна економіка, дослідження транзакцій, статистика та теорії прийняття рішень. SPPR може використовувати методи штучного інтелекту для вирішення геристичних проблем, нерозчинних формальними методами. Належне використання інструментів підтримки прийняття рішень підвищує продуктивність та ефективність, а також дає бізнесу порівняльну перевагу передконкурентами, що дозволяє їм приймати оптимальні рішення для процесів та їх параметрів, для операцій з бізнес-планування, логістики або інвестицій.

Для аналізу предметної зони та формування рішення СПРП може використовувати різні методи, такі як пошук інформації, знань або шаблонів у базах даних, обґрунтування на основі кейсів, ситуаційне моделювання та ситуаційний аналіз, інтелектуальний аналіз інформації, моделювання

суб'єктивних сприйняття експертів про ситуацію, нейронні мережі, генетичні алгоритми тощо.

Інтелектуальна SPPR, або ISPR, називається системою підтримки прийняття рішень, побудованою на основі алгоритмів штучного інтелекту. Завдяки цим МОП можна знайти оптимальне рішення навіть при багатокритичних, низькоструктурованих або неструктурованих завданнях.

Класифікація SPR

У зв'язку з відсутністю суворого єдиного визначення систем підтримки прийняття рішень, ми також стикаємося з відсутністю звичайної класифікації. Найбільш загальна класифікація рейтингу SPR відповідає основним операціям, які вони виконують, від систем,орієнтованих на дані, до модельно-орієнтованих систем (Alter S. L. 1980 р.). Ця класифікація була заснована на аналізі 56 різних CPD і включає в себе наступні сім типів:

- Файлова система, яка надає користувачеві легкий доступ до елементів даних.
- Системи аналізу даних, які дозволяють обробляти дані за допомогою автоматизованих додатків, які спеціалізуються на конкретних завданнях або загальних методах.
- Системи аналізу даних, які забезпечують доступ до ряду баз даних, орієнтованих на пошук, і невеликих моделей.
- Бухгалтерські та фінансові моделі, метою яких є х, полягає в тому, щоб побачити наступні можливі дії.
- Візуальні (репрезентативні) моделі, які оцінюють наслідки дій на основі симуляційних моделей.
- Моделі оптимізації, які спрямовані на розробку основних поведінковихпринципівшляхом створення найкращого рішення відповідно до ряду обмежень.
- Поради моделей, які виконують логічну обробку, щоб знайти конкретне рішення для добре структурованої або глибоко вивченої завдання.

Слід зазначити, що, виходячи з досліджень, обсяг інформації, виробленої людством, більш ніж подвоюється протягом двох років. В результаті стає все важче знайти і отримати правильні дані із загального інформаційного потоку. Для того, щоб впоратися з цим завданням, зараз активно розвивається нова сфера видобутку даних, яка прагне вирішити проблему пошуку нових знань, необхідної інформації та взаємного впливу різних факторів у великих обсягах неструктурованих даних. Також для вирішення проблеми вилучення інформації з величезних обсягів неструктурованих даних значного різноманіття активно розвивається новий клас значного різноманіття підходів, методів та інструментів - «BigData» (Big Data). Цей клас контрастує зі стандартними системами управління базами даних за допомогою інструментів масової паралельної обробки, які постійно збільшуються в масштабі неструктурованих даних.

Одним з найскладніших і трудомістких процесів є процес здобуття знань, велика залежність ситуацій від контексту, наличие противоречивости и неполноты имеющихся данных, різнотипність неоднозначність, ненадійність і невизначеність інформації[10] . Виходячи з вищесказаного ,інтелектуальні методи Data Mining Data Mining для вирішення існуючих проблем є одним з перспективних рішень. У рамках межах області видобутку даних дослідники розрізняють шість різних типів завдань:

- Пошук залежностей
- Класифікація за групами,
- Виявлення аномалій і відхилень,
- регресійний аналіз,
- Узагальнення
- пошук шаблонів підсекцій у послідовності даних.

Актуальність прогнозування часових рядів

Одним з найбільш перспективних і складних завдань є прогнозування ***часовий ряд*** Завдання прогнозування часового ряду знаходяться в багатьох сферах життя людини. Я перерахую деякі з них:

1) Діагностичні завдання в медицині. Прикладами є вік, стать. Об’єктами даних задач є пацієнти. Зробити краще лікування

- Передбачити тривалість і результат захворювання
- Оцінити ризик ускладнень
- виявити синдроми - найпоширеніші для цього захворювання (сукупність симптомів).

Перевага таких систем над лікарями полягає в тому, що вони дозволяють швидко провести аналіз і узагальнення великої кількості прецедентів, а також прийняти обґрунтоване рішення на їх основі.

Все більшою популярністю користуються автоматизовані торгові стратегії - алгоритми, які дозволяють приймати торгові рішення без участі людини. На даний момент багато банків і підприємств вже використовують автоматизовані торгові системи (торгові роботи) для отримання доходу для своїх власників. Метою цих систем є спрощення роботи гравців фондового ринку, а в подальшому і в повному обсязі взяти на себе здійснення торгових операцій.

Деякі параметри цих завдань можуть бути невизначені,встановлені неточно або як інтервал значень.

Деякі об'ємні показники можуть змінюватися кілька разів за відносно короткий проміжок часу.

Логічно використовувати значення ***інтервалу*** - не одне значення - для роботи з такими даними, а безперервний набір значень між парою значень, що встановлює межі інтервалу. Інтервал характеризується властивістю, яка разом з будь-якими двома числами містить будь-яке між ними.

Для деяких завдань інтервал також є рішенням, наприклад, прогнозування зміни цінових пропозицій, оптимального режиму роботи або температурного діапазону.

Класифікація часовий ряд

Існує кілька різних класифікацій часового ряду. Давайте подивимося на основні з них:

1. За характером перегляду вимірювань:

- Середньогабаритні ряди;
- Відносні показники
- абсолютні показники.

2. За типом параметрами часу:

- Миттєвий часовий ряд. За часову серію цього виду вимірювань відповідають за стан спостережуваних значень в певні моменти часу.
- Інтервальні часові ряди.

3. За тривалістю проміжку між моментами часу, які включають вимірювання:

- Повні (рівні) - коли вимірювання слідують один за одним через рівні проміжки часу або моменти початку і кінця вимірювань завжди відокремлюються один від одного з рівним проміжку часу;
- неповні (нерівномірні) - якщо описані вище умови не виконані.

4. До речі, дані формуються, часовий ряд ділиться на:

- детерміновані - які є значеннями якоїсь невипадкової функції час від часу (серія інформації, замовленої за місяцем у днях);

5. Залежно від основної тенденції:

- Стаціонарні ряди з часом не змінюють своїх основних характеристик (математичне очікування перманентності и і функція автокореляції процесу залежить тільки від розміру інтервалу між вимірами);
- не стаціонарний - не відповідає вищевказаному критерію.

Коли ми говоримо про передбачення часових рядів, в цій роботі йдеться про завдання вивчення і аналізу часового ряду спостережень за деякими параметрами навчального процесу для навчання на основі інтервалної нейронної мережі.

1.2 Аналіз загальних методів підтримки прийняття рішень

Для аналізу великого обсягу інформації, прогнозування поведінки керованої системи та розробки контрольного рішення були розроблені та використані різні методи. Давайте подивимося на основні з них:

1. Пошук інформації - процес пошуку серед великої кількості текстів або документів таких, що містять інформацію по предмету, що цікавить, відповідають заданій умові пошуку або містять необхідні дані, факти, інформацію .

Зазвичай пошук інформації складається з трьох кроків:

- Формулювання критеріїв пошуку та створення запиту;
- Ідентифікувати документи, які можуть містити інформацію, що цікавить;
- Детальний пошук у вибраних документах та оцінка результатів пошуку.

2. Дерево прийняття рішень - є графіком з одним початковим піком. На краях яких написані атрибути, від яких залежить цільова функція.

3. Кластерний аналіз - це набір різних алгоритмів класифікації, що дозволяє на основі подібності приватних рис або набору ознак розбити на окремі групи спостережуваних наборів знаків (ситуацій). Якщо алгоритм кластеризації правильно застосовується, після того, як він працює, наступне твердження вірно - для будь-якого набору рис, набір, який трапляється з ним в одному кластері більше схожий на будь-який набір з іншого , кластера з ним. А значит и решение

4.Прецедентний Р-1 –це підхід, який передбачає вирішення нових завдань за допомогою та/або адаптації рішень до вже відомих завдань.

Як правило, рішення в даному випадку складається з чотирьох кроків:

- Вибір подібного прецеденту з бази даних;
- Адаптація рішення знайденого прецеденту до проблеми, яку необхідно вирішити;
- Перевірка рішення.

- Додайте до бази даних відомості про вирішену проблему разом із її вирішенням.

5. Моделювання - це метод, спрямований на вивчення спостережуваної реальної системи шляхом заміни її на досить точну модель. .

6. Генетичні алгоритми - алгоритми вирішення складних неформальних завдань, при відсутності кожного наступого набору використовуються при дуже великих розмірах завдань і при відсутності і впорядкування в оригінальних даних. Ці алгоритми імітують еволюційну теорію Дарвіна шляхом пошуку рішення шляхом послідовного вдосконалення наборів потенційних

7. Когнітивність

У даній моделі інформація про систему представлена як набір понять і причинно-наслідковий зв'язок мережі, що їх пов'язує, називається когнітивною картою. Для отримання найкращої стратегії управління ми використовуємо методи аналітичної обробки для дослідження структури системи та прогнозування її поведінки в різних стратегіях управління.

8. Методи статистичного прогнозування – це математичні методи прогнозування, які роблять прогнози на основі об'єктивних даних про спостережуваний процес.

Статистичні методи поставили перед собою завдання знайти якийсь функціонал зі спостережуваних параметрів процесу, значення якого були б прогнозом. Ця функція вимагає мінімізації математичного очікування квадрата різниці між його значенням в якийсь момент часу і реальною вартістю прогнозованого значення.

9. Нейронні мережі

Головною перевагою нейронних мереж є вміння вчитися. Через це нейронні мережі здатні автоматично здобувати знання, але не в змозі пояснити ці знання. Навчена нейронна мережа з'являється користувачеві як якась чорна скринька.

Однією з найпопулярніших моделей нейронної мережі є багатошаровий персептрон, заснований на взаємодії шарів штучних нейронів.

Для підготовки багатошарової персептичної, використовується алгоритм зворотного поширення помилки, заснований на методі градієнтного спуску.

10. Системи з нечіткою логікою - системи, які працюють з мовними змінними і апаратом нечіткої логіки, що дозволяє прийняття функції значення приналежності в інтервалі від 0 до 1 і логічних операцій над неправильними наборами.

Системи з нечіткою логікою добре підходять для пояснення результатів, які вони дають, але не можуть отримати знання, за якими потрібно прийняти рішення.

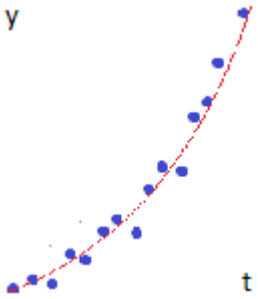
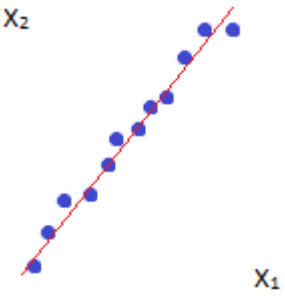
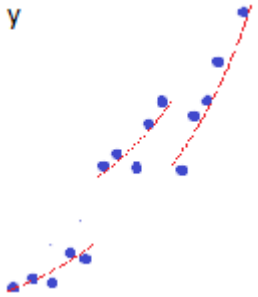
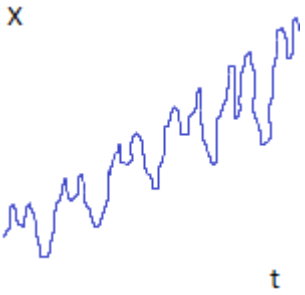
Системи з нечіткою логікою мають сенс використовувати в наступних випадках:

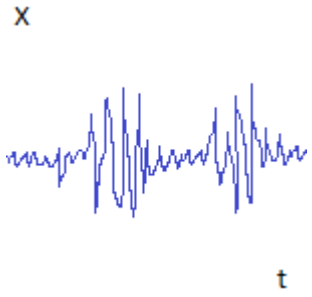
- Для складних процесів, за відсутності інших стандартних математичних моделей;
- якщо знання є експертними і про процеси або систему можуть бути сформульовані тільки в мовній формі.

Головний недолік систем з нечіткою логікою полягає в тому,що набір нечітких правил і типу функції приналежності суб'єктивно встановлюється людиною-експертом і може бути некоректним або неповним.

11.Методи прогнозування експертів – це методи, які покладаються на прогнозування оцінок людських експертів.

1.3 Актуальність створення алгоритму прогнозування часового ряду.

Статистичний метод	Передумови для нанесення	Приклад моделі	Зразки даних
Екстраполярні методи прогнозування	Існує тенденція або довгострокова тенденція в часовому ряду	y -0 - 1₁т- 2тt² ... aₙn немає^{n}ε	
Методи аналізу кореляційних даних	Існує кореляція між значеннями різних часових рядів	$X_1 = \mu_1 + \rho\frac{\sigma_1}{\sigma_2}(X_2 - \mu_2)$	
Моделі, в параметри яких закладені один з видів втручання (стійкий стрибок, стійкий поступовий, стрибки тимчасові)	Часовий ряд з різкими змінами тенденцій процесу під зовнішнім або внутрішнім впливом (зазвичай зовнішній)	y =0 +1т+2т2+2 w*1{t1<t<t2} (staid polyne з стрибком скроневого ефекту)	
Гармонійні моделі або моделі середнього прокату автора	Часовий ряд має постійні коливання тренду з невідомим періодом на початку дослідження	$\Delta^d X_t = c + \sum_{i=1}^{p} a_i \Delta^d X_{t-i} + \varepsilon_t + \sum_{j=1}^{q} b_j \varepsilon_{t-j}$ ARIMA (p,d,q)	

Статистичний метод	Передумови для нанесення	Приклад моделі	Зразки даних
Garch модельний	Спостереження з великими і малими відхиленнями від середніх, як правило, утворюють кластери	$y_t = x'_t \beta + u_t$ $\sigma_t^2 = \alpha_0 + \sum_{i=1}^{p} \alpha_i u_{t-i}^2 +$ $+ \sum_{j=1}^{q} \gamma_j \sigma_{t-j}^2$ GARCH(c,q)	

Таблиця 1 Поширені статистичні методи прогнозування часовий ряд

Ось короткий порівняльний опис методів, які дозволяють викопати оцінки параметрів або зробити прогнозування, виключивши решту з розгляду:

Метод	Вимога до кваліфікації користувача	Можливість витягувати знання з власних даних	Остаточна кількість можливих рішень	Швидкість аналізу інформації та прийняття рішень
Дерева рішень	Невеликі	Відсутні	Так	Низьким
Аналіз кластера	Середній	Присутній	Так	Висока
Міркування на основі інциденту	Середній	В невеликій мірі	Так	Висока
Методи статистичного прогнозування	Великий	В невеликій мірі	Ні	Низьким
Нейронні мережі	Невеликі	Присутній	Ні	Висока
Експертні методи	Середній	Присутній	Ні	Низьким

Таблиця 2 Порівняння методів прогнозування часів

Розглядаючи існуючі методи, можна прийти до наступних висновків:

- Існуючі методи мають труднощі з пошуком змінних, які мають найбільший вплив на прогнозоване значення.
- Рідкісний метод s, крім статистики, дозволяє працювати з невизначеними даними.
- Немає просто у використанні універсального (не предмета) методу, який вимагає невеликої кількості часу

1.4 Налаштування завдання

Розглядаючи завдання підтримки прийняття рішень за допомогою методів прогнозування або оцінки, а також існуючих підходів до її вирішення, можна сформулювати наступні вимоги до оболонки:

1. Алгоритм, за яким буде працювати оболонка SPPR, повинен забезпечити максимальну точність при прогнозування інтересу значення при наявності складних неочевидних залежностей між параметром.
2. Слід працювати з невизначеними даними.
3. Продуктивність остаточної реалізації повинна бути достатньою для прогнозування оцінок часових рядіи не з використанням суперкомп'ютерів або обчислювальних кластерів (комп'ютерних груп).

Таким чином, виходячи з вимог до оболонки SPPR, ми підійдемо до наступного набору завдання:

1. Змінити алгоритм поширення помилок для навчання інтервалу нейронних мереж, які дозволяють працювати з інтервальними даними.
2. Змінити генетичний алгоритм для визначення оптимальної структури інтервалу нейронної мережі і набору параметрів, на основі яких потрібно робити прогнозування, а також обсягу інформації про ці параметри для оперативного і точного рішення.
3. Впровадити розроблений алгоритм за допомогою паралельної обчислювальної архітектури на GPus of the HellaPter, щоб скоротити витрати на час.
4. Застосувати отриману систему до реальних завдань прогнозування, щоб порівняти з існуючими системами і методами.

2 ЯК РЕАЛІЗУВАТИ АЛГОРИТМИ ПРОГНОЗУВАННЯ ЧАСОВОГО РЯДУ

2.1 Використання інтервальної нейронної мережі

Стаття доводить, що будь-яка безперервна n змінної функції n може бути приблизна нейронними мережами за допомогою будь-якої функції безперервної активації однієї змінної.

Як модель складної багатовизначної нелітерної регресії нейронна мережа потенційно перевершує за точністю до перерахованих вище методів, а також має ряд переваг:

(a) Можливість роботи з шумом і неінформативними входами - тільки нейронна мережа може визначити їх непридатність для вирішення проблеми і явно відкинути їх, обнулення відповідних коефіцієнтів.

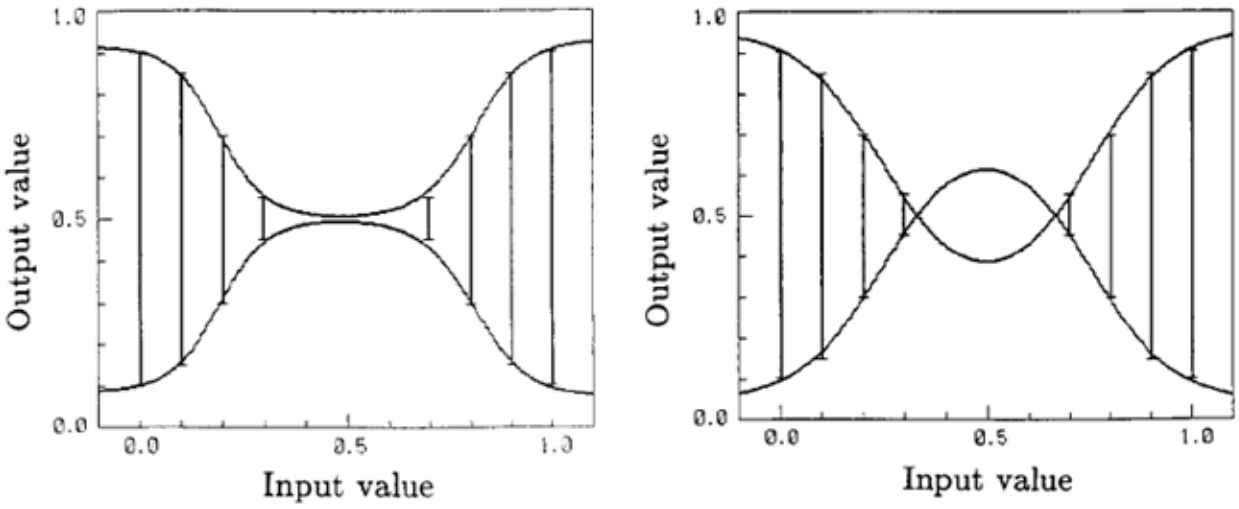
(b) Можливість вирішувати кілька проблем одночасно, якщо нейронна мережа має кілька виходів.

Наприклад, можна підключити нову нейронну мережу і навчити її таким чином, щоб був вхід для першої нейронної мережі. Нейронна мережа має менше вимог до кваліфікації користувача, ніж, наприклад, складні статистичні моделі, здатні давати аналогічні результати.

(c) Спочатку встановивши синаптичні ваги нейронної мережі, ви можете відтворити і перевірити передбачувані статистичні закономірності, а також поліпшити їх, навчивши мережу.

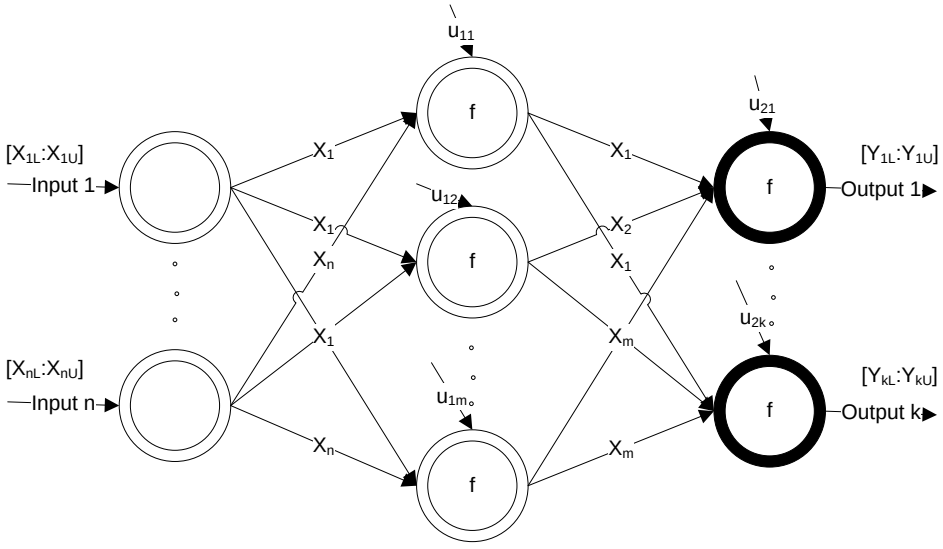
Інтервальні нейронні мережі

У разі прогнозування інтервалу необхідно використовувати інтервальні нейронні мережі.



Малюнок 1. Лівий інтервал нейронної мережі, праворуч дві стандартні нейронні мережі. Вертикальні лінії зображують інтервали вихідного дня тренувальних наборів.

Інтервальна нейронна мережа **-** це система підключених і взаємодіючих інтервальних нейронів, які мають при вході і виводі значення, встановлені у вигляді інтервалу (не одне значення, а континуумний набір значень між парою значень, що встановлює межі інтервалу).



Малюнок 2

2 n m k 2 '2' (n) м, k, k, K і т.д. – число вхідних нейронів, –число прихованих нейронів,

Кожному з нейронів у вході надається інтервативне значення, на підставі якого планується отримати прогноз. Для усіх інших нейронів, на виході отримуємо інтервальні значення на основі формули: нейрон

$$Y = f\left(\sum_{i=1}^{N} w_i X_i + u_i\right),$$

Вираховується з правилами інтервальної аріфметики :, де *Ю i* - вихідні, *X и-* у кожного нейрона , *ф i* - *u Y i* функціямі, - функція актівації, *N* - готельний об'єкт

$$A * B = [a_L, a_U] * [b_L, b_U] =$$

$$= [\min\{ a * b \mid a \in [a_L, a_U], b \in [b_L, b_U] \}, \max\{ a * b \mid a \in [a_L, a_U], b \in [b_L, b_U] \}]$$

де - будь-яка операція. * Наприклад:

$$A + B = [a_L, a_U] + [b_L, b_U] = [a_L + b_L, a_U + b_U],$$

$$A - B = [a_L, a_U] - [b_L, b_U] = [a_L - \max\{ b_L, b_U \}, a_U - \min\{ b_L, b_U \}],$$

$$A \cdot w = [a_L, a_U] \cdot w = \begin{cases} [a_L \cdot w, a_U \cdot w], & w \geq 0 \\ [a_U \cdot w, a_L \cdot w], & w < 0 \end{cases},$$

$$A \cdot B = [a_L, a_U] \cdot [b_L, b_U] = [\min\{ a_L b_L, a_U b_U, a_L b_U, a_U b_L \}, \max\{ a_L b_L, a_U b_U, a_L b_U, a_U b_L \}].$$

Використання інтервативних значень дозволяє прогнозувати та використовувати як входи:

- Стандартні параметри інтервалу (ціна відкриття/закриття, купівля/продаж),
- Інтервальні значення, які зменшують кількість входів (замість хвилинного курсу дозволяють мережі вводити від мінімальної до максимальної ціни за годину або день),
- Значення, значення яких спотворені помилками округлення через неточність вимірювальних приладів
- Випадкові значення (прийом значень з інтервалом).

- Звичайні - не інтервальні параметри, прирівнюючи нижню і верхню межу інтервалу.

На основі статей «22, 23» розроблено алгоритм вивчення інтервальної мережі(узагальнення алгоритму зворотного розподілу похибки):

Давайте позначимо входи нейронної мережі як приховані виходи шарів - де $I_i = [I_i^L, I_i^U]$ $H_j = f(Net_j) f$ є функцією активації, а вихідні виходи - де; цільові значення.

$$Net_j = \sum_{i=1}^{num\_inp} w_{ji} \cdot I_i + \theta_j = \left[\sum_{\substack{i=1 \\ w_{ji} \geq 0}}^{num\_inp} w_{ji} \cdot I_i^L + \sum_{\substack{i=1 \\ w_{ji} < 0}}^{num\_inp} w_{ji} \cdot I_i^U + \theta_j^L ; \quad \sum_{\substack{i=1 \\ w_{ji} \geq 0}}^{num\_inp} w_{ji} \cdot I_i^{/U} + \sum_{\substack{i=1 \\ w_{ji} < 0}}^{num\_inp} w_{ji} \cdot I_i^L + \theta_j^U \right]$$

$$O_k = f(Net_k) \qquad Net_k = \sum_{j=1}^{num\_hid} w_{kj} \cdot H_j + \theta_k =$$

$$= \left[\sum_{\substack{j=1 \\ w_{kj} \geq 0}}^{num\_hid} w_{kj} \cdot H_j^L + \sum_{\substack{j=1 \\ w_{kj} \geq 0}}^{num\_hid} w_{kj} \cdot H_j^U + \theta_k^L ; \quad \sum_{\substack{j=1 \\ w_{kj} \geq 0}}^{num\_hid} w_{kj} \cdot H_j^U + \sum_{\substack{j=1 \\ w_{kj} \geq 0}}^{num\_hid} w_{kj} \cdot H_j^L + \theta_k^U \right] T_k = [t_k^L, t_k^U]$$

Функція помилки: . $E = \frac{1}{2} \sum_{k=1}^{num\_out} \left(\left(t_k^L - o_k^L \right)^2 + \left(t_k^U - o_k^U \right)^2 \right)$

Зміна ваги відбувається $\Delta w_{ji}(t) = \eta \left(-\frac{\partial E}{\partial w_{ji}} \right) + \beta \Delta w_{ji}(t-1)$ де , за правилом: де - коефіцієнт тренування - фактор η пульсу, і розраховується за наступними β розрахунками: $\partial E / \partial w_{ji}$

Між прихованим шаром і виходом, $w_{kj} \geq 0$ якщо:

$$\partial E / \partial w_{kj} = \frac{\partial}{\partial w_{kj}} \left\{ \left(t_k^L - o_k^L \right)^2 / 2 \right\} + \frac{\partial}{\partial w_{kj}} \left\{ \left(t_k^U - o_k^U \right)^2 / 2 \right\} =$$

$$= \frac{\partial}{\partial o_k^L} \left\{ \left(t_k^L - o_k^L \right)^2 / 2 \right\} \frac{\partial o_k^L}{\partial net_k^L} \frac{\partial net_k^L}{\partial w_{kj}} + \frac{\partial}{\partial o_k^U} \left\{ \left(t_k^U - o_k^U \right)^2 / 2 \right\} \frac{\partial o_k^U}{\partial net_k^U} \frac{\partial net_k^U}{\partial w_{kj}} =$$

$$= -\left(t_k^L - o_k^L \right) o_k^L \left(1 - o_k^L \right) h_j^L - \left(t_k^U - o_k^U \right) o_k^U \left(1 - o_k^U \right) h_j^U ,$$

Якщо: $w_{kj} < 0$

$$\partial E / \partial w_{kj} = -\left(t_k^L - o_k^L \right) o_k^L \left(1 - o_k^L \right) h_j^U - \left(t_k^U - o_k^U \right) o_k^U \left(1 - o_k^U \right) h_j^L ;$$

$w_{ji} \geq 0$ Між входом і прихованим шаром, якщо:

$$\partial E / \partial w_{ji} = \frac{\partial}{\partial w_{ji}} \left\{ \left(t_k^L - o_k^L \right)^2 / 2 \right\} + \frac{\partial}{\partial w_{ji}} \left\{ \left(t_k^U - o_k^U \right)^2 / 2 \right\} =$$

$$= \frac{\partial}{\partial o_k^L} \left\{ \left(t_k^L - o_k^L \right)^2 / 2 \right\} \frac{\partial o_k^L}{\partial net_k^L} \left[\sum_{k=1, w_{kj} \geq 0}^{num\_out} \frac{\partial net_k^L}{\partial o_{kj}^L} \frac{\partial o_{kj}^L}{\partial net_{kj}^L} \frac{\partial net_{kj}^L}{\partial w_{ji}} + \sum_{k=1, w_{kj} < 0}^{num\_out} \frac{\partial net_k^L}{\partial o_{kj}^U} \frac{\partial o_{kj}^U}{\partial net_{kj}^U} \frac{\partial net_{kj}^U}{\partial w_{ji}} \right] +$$

$$+ \frac{\partial}{\partial o_k^L} \left\{ \left(t_k^U - o_k^U \right)^2 / 2 \right\} \frac{\partial o_k^U}{\partial net_k^U} \left[\sum_{k=1, w_{kj} < 0}^{num\_out} \frac{\partial net_k^U}{\partial o_{kj}^U} \frac{\partial o_{kj}^U}{\partial net_{kj}^U} \frac{\partial net_{kj}^U}{\partial w_{ji}} + \sum_{k=1, w_{kj} \geq 0}^{num\_out} \frac{\partial net_k^U}{\partial o_{kj}^L} \frac{\partial o_{kj}^L}{\partial net_{kj}^L} \frac{\partial net_{kj}^L}{\partial w_{ji}} \right] =$$

$$= - \sum_{k=1, w_{kj} \geq 0}^{num\_out} \left(\left(t_k^L - o_k^L \right) o_k^L \left(1 - o_k^L \right) w_{kj} \right) h_j^L (1 - h_j^L) I_i^L - \sum_{k=1, w_{kj} < 0}^{num\_out} \left(\left(t_k^L - o_k^L \right) o_k^L \left(1 - o_k^L \right) w_{kj} \right) h_j^U (1 - h_j^U) I_i^U -$$

$$- \sum_{k=1, w_{kj} \geq 0}^{num\_out} \left(\left(t_k^U - o_k^U \right) o_k^U \left(1 - o_k^U \right) w_{kj} \right) h_j^U (1 - h_j^U) I_i^U - \sum_{k=1, w_{kj} < 0}^{num\_out} \left(\left(t_k^U - o_k^U \right) o_k^U \left(1 - o_k^U \right) w_{kj} \right) h_j^L (1 - h_j^L) I_i^L ,$$

если $w_{ji} \geq 0$:

$$\partial E / \partial w_{ji} = - \sum_{k=1, w_{kj} \geq 0}^{num\_out} \left(\left(t_k^L - o_k^L \right) o_k^L \left(1 - o_k^L \right) w_{kj} \right) h_j^L (1 - h_j^L) I_i^U - \sum_{k=1, w_{kj} < 0}^{num\_out} \left(\left(t_k^L - o_k^L \right) o_k^L \left(1 - o_k^L \right) w_{kj} \right) h_j^U (1 - h_j^U) I_i^L -$$

$$- \sum_{k=1, w_{kj} \geq 0}^{num\_out} \left(\left(t_k^U - o_k^U \right) o_k^U \left(1 - o_k^U \right) w_{kj} \right) h_j^U (1 - h_j^U) I_i^L - \sum_{k=1, w_{kj} < 0}^{num\_out} \left(\left(t_k^U - o_k^U \right) o_k^U \left(1 - o_k^U \right) w_{kj} \right) h_j^L (1 - h_j^L) I_i^U .$$

2.2 Використання генетичних алгоритмів

Якщо ви надішлете значення всіх відомих параметрів на вхід нейронної мережі, це значно уповільнить швидкість її роботи, а також збільшить ймовірність того, що мережа дійсно не залежить від залежностей. З іншого боку, помістіть лише частину параметрів, які здаються тим, хто приймає рішення, найбільш значущими, можна не помітити параметри, які фактично сприяють значенню прогнозованого значення.

Для вирішення цієї проблеми розроблено кілька алгоритмів. Найпопулярнішим алгоритмом для використання в нейронних мережах є алгоритм, заснований на послідовному збільшенні кількості вхідних параметрів шляхом додавання найбільш значущих, при цьому додавання нового параметра дасть помітний результат. Перший крок алгоритму тренує власну мережу єдиним входом для кожного параметра. Варіант, на основі якого мережа дає найменшу похибку в прогнозування,вважається найбільш значущим і додається до кількості значних. Алгоритм повторюється, доки не буде додавання нового значущого параметра зменшуючого помилку прогнозування.

Кореляційний аналіз не підходить як метод обробки вхідних параметрів для вирішення проблеми, оскільки кореляція відображає лише лінійну залежність значень, але не відображає їх функціональний зв'язок.

Як метод вирішення цієї проблеми ми вибрали генетичний алгоритм. ***Він*** заснований на генетичних процесах біологічних організмів: біологічні популяції розвиваються протягом декількох поколінь за умови законів природного відбору і принципу «виживання найбільш придатних», виявлених Чарльзом Дарвіном. все чаще использу ***алгоритм*** - адаптивны й Так само генетичні алгоритми Найбільш підходящі особи відбираються для створення новогово покоління (відтворення їжі потомства(а) з понтипами перехресного перетину з іншими особами населення. Кожне нове покоління, відносно попереднього покоління,

містить велику кількість характеристик, якими володіють хороші члени попереднього покоління.

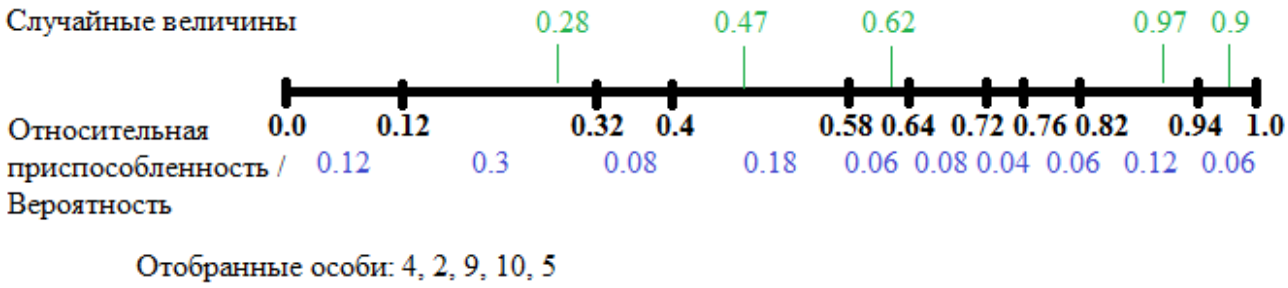
Метод	Пошук лінійної залежністі	Пошук функціональної залежності	Швидкість отримання перших значущих результатів
Послідовне збільшення кількості налаштувань	Так	Так	Низька
Кореляційний аналіз	Так	Ні	Висока
Генетичний алгоритм	Так	Так	Висока

Таблиця 1

Більш детально охарактеризувати принцип генетичного алгоритму взаємодії з нейронними мережами.

Початкові покоління параметрів (індивідуумів) для генетичного алгоритму визначаються випадковим чином. Крім того, найбільш підходящими особами вважаються набори, навчені, на яких мережі дають мінімальні помилки. Нове покоління індивідуумів виходить шляхом перетину найбільш адаптованих особин попереднього покоління і мутації.

Підбір особини для взаємопов'янення здійснюється за допомогою «вибору рулетки». Цей вид відбору є мовірнісним методом випадкового підбору особ, де кожна людина має попередню ймовірність бути обраною, корелює зі значенням своєї функції фітнесу. Кожна людина поставлена у відповідність з відносною придатною.Необхідна кількість разів запуску безперервного генератора рівномірно розподіляється з інтервалом 0,1 (базовевипадкове значення). Цей метод проілюстрований на рисунку 3.



Малюнок 3 Ілюстрація вибору рулону

Це дозволяє періодично розбавляти генофонд і, завдяки цьому, запобігати зближенню до місцевої екстремальної цільової функції.

Перетин (кросовер) відбувається наступним чином - випадково визначена точка всередині хромосоми (набір параметрів) називається точкою розриву, в якій обидві хромосоми діляться на дві частини. На кожному циклі, щоб запобігти передчасному зближенню, з деякою низькою ймовірністю відбувається мутація особ, яка полягає в зміні окремого генно-параметра в особистості на інший із загального набору параметрів.

Малюнок 4 Ілюстрація процедури кросовера

Малюнок 5 Ілюстрація роботи генетичного алгоритму

Отримання нових поколінь триває до тих пір, поки населення не зійшло або не перевищить призначену користувачем максимальну кількість поколінь.

Давайте використовувати принцип генетичного алгоритму для роботи над наступним простим прикладом:

Розглянемо завдання пошуку хромосоми, яка має максимальну кількість одиниць. Розглянемо вирішення цієї проблеми за допомогою генетичного алгоритму.

Ми сформуємо первісне населення за допомогою будь-якого із запропонованих методів:

ch_1 - [010111101010]; ch_2 - [110000101101];

ch_3 - [11000001001]; ch_4 - [100010000001];

ch_5 - [001101011000]; ch_6 - [10001000101];

ch_7 - [001000001010]; ch_8 - [110011010110].

Оценим приспоспонітии хромосом в созданной популяции. Так как в нашем простом мы и таку ю хромосокий,которая содержите обшивка етинац. Следовательно, функцияприлежня должна за заее стецть отства еднац в хромососоме. Підм обсимозна функция бєдлносности атиности *Ф*. А за ееосие брать сумюхов хромососомы. В этомслу ичасыя для каждой хромососомый из полу щуеип иой популяции хлт следующые:

F((ч.$_1$) -7; *F*(чch$_2$) -6;

F((ч$_3$) -5; *F*(ч.ch$_4$) -3;

F((ch$_5$) -5; *F*(chch$_6$) -4;

F((ч.$_7$) -3; *F*(chch$_8$) -7.

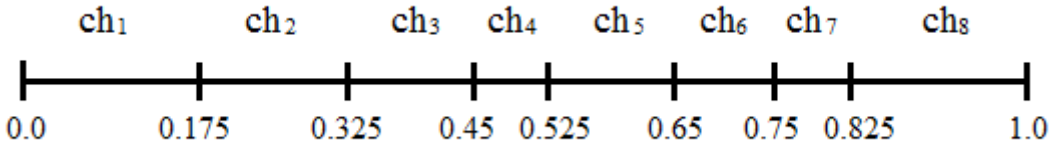
Вибір виконаний рулонним методом

v(ch$_1$) =0,175; *v*(ch$_2$) =0,15;

v(ch$_3$) =0.125; *v*(ch$_4$) =0,075;

v(ch$_5$) =0.125; *v*(ch$_6$) =0,1;

v(ch$_7$) =0,075; *v*(ch$_8$) =0.175.



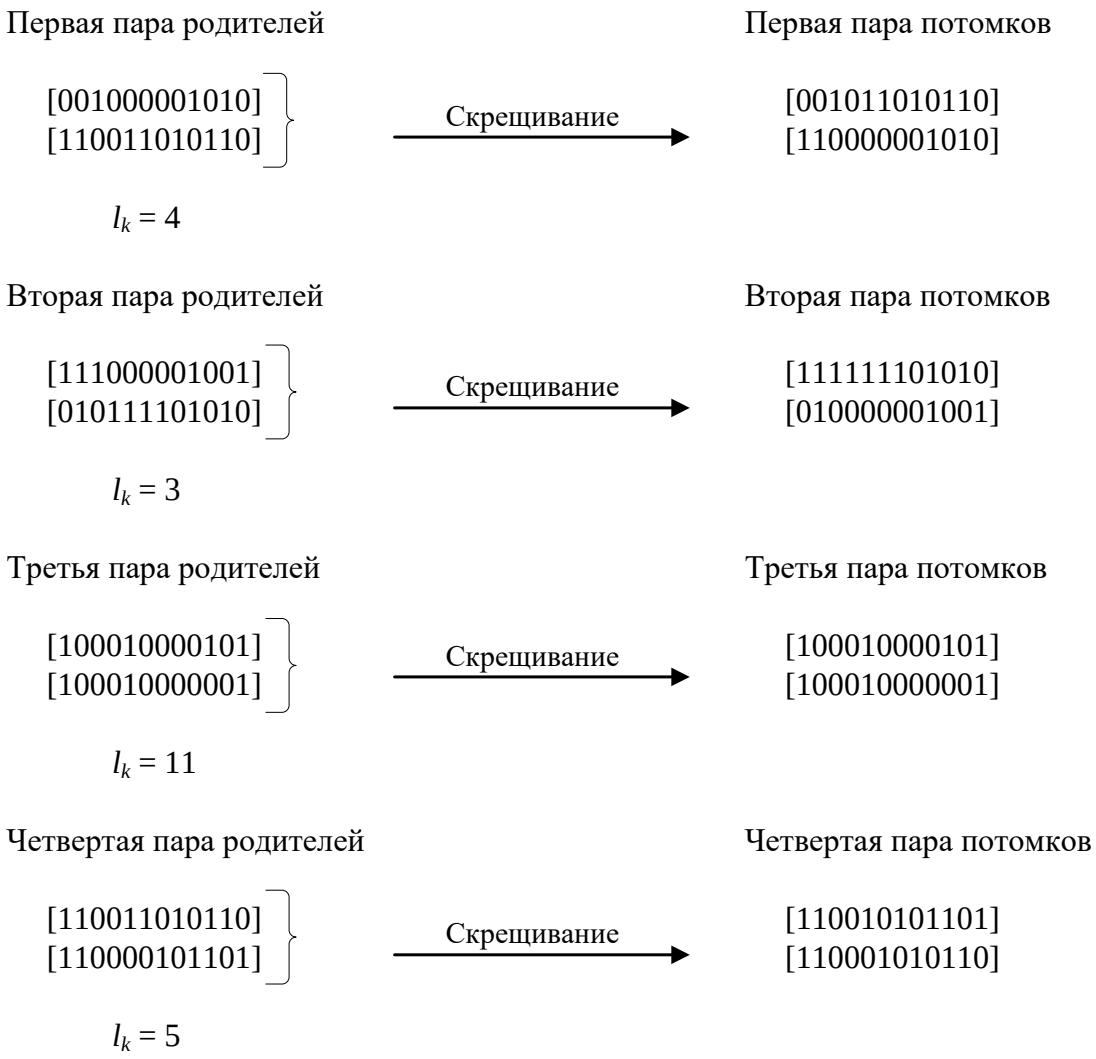
Малюнок 5 Розділіть виріз для вибору рулону

Креслення з вбудованим сплітом зводиться до випадкового вибору числа що вказує відповідний сектор сегмента,тобто відповідного сегмента специфічної хромосоми.

Давайте грати 8 випадкових чисел:

0,79 0,44 0,9 0,74 0,04 0,86 0,48 0,23

Використання перехреєсних операторів. У цьому прикладі ми не будемо піддавати хромосоми, вибрані для відбору, мутаціям,і, все додамо до популяції хромосом для переплетення. Чотири пари батьків сформували чотири пари батьків з цих хромосом.



Малюнок 6 Хромосомний процес перетину

Ch_1 - [001011010110]; Ch_2 - [11000001010];

Ch_3 - [111111101010]; Ch_4 - [010000001001];

Ch_5 - [10001000101]; Ch_6 - [100010000001];

Ch_7 - [110010101101]; Ch_8 - [110001010110].

Для того, щоб відрізнити щойно отримані хромосоми від хромосом попереднього населення, ми почнемо їх позначення з великої літери С.

За генетичним алгоритмом ми повертаємося до другого етапу - оцінки придатності хромосом від новоствореного населення.

$F(CCh_1)$ -6; $F(CCгод_2)$ -4;

$F(CCh_3)$- 9; $F(CCh_4)$- 3;

$F(CCh_5)$- 4; $F(CCh_6)$- 3;

$F(CCh_7)$ -7; $F(CCh_8)$ -6.

Видно, що популяція нащадків має середню функцію фітнесу вище, ніж у батьківського населення. шанс ов хромосомы с большими значениями функции приспособленности име ют больше Зверніть увагу, що після перетину була отримана хромосома Ch 3 значення функції фітнесу, яка більше, ніж 3 у будь-якої хромосоми від батьківського населення.

2.3 Використання паралельної обчислювальної архітектури на графічних процесорах

Розрахунки по ГПУ (ГПУ) складаються з використання процесора (ЦП) спільно з графічним процесором для прискорення обчислень шляхом масштабного паралелі алгоритмів. Цей метод обчислень був винайдений більше десяти років тому і зараз активно використовується для вирішення широкого спектру завдань, які вимагають швидкого виконання громіздких розрахунків.

Хоча ядра ГПУ не такі високо емоційні,я к ядро процесора, перевага ГПУ досягається за кількістю з них (від близько 300 ядер на стандартних відеокартах, і до більш ніж 4000 ядер на одній з останніх розробок ASUS).

Один з найдорожчих продуктів домашнього процесора Intel Corei7-975 XE 3,33 ГГц 2009, пропонує пікову продуктивність 53.3 GFlops, так як відеокарта Nvidia Tesla K20X (2688 ядер) за допомогою паралельних обчислень може запропонувати теоретичну пікову продуктивність 3.95 TFlops q26q. А вартість відеокарти набагато менше, ніж на продуктивність кластера від процесора.

Недоліком при використанні графічного процесора для обчислень є відсутність ядер комп'ютерної відеокарти, відносно невелика швидкість копіювання даних з оперативної пам'яті в пам'ять графічного процесора, а також складність масово паралелі деяких алгоритмів для одночасного використання всіх доступних ядер.

Архітектури	Cuda	Opencl	Прямий комп'ютер	Amd	C++ AMP
Спеціалізація на відеокарти картах	Nvidia	Hi	Hi	Hi	Hi
Мови програмування	C, C++, Фортран	C	C	C	Cj

Архітектури	Cuda	Opencl	Прямий комп'ютер	Amd	C++ AMP
Кросплатформний	Так	Так	Windows	Так	Windows
Можливість використання масивів змінної довжини	Так	Ні	Так	Ні	Так
Архітектури	Cuda	Opencl	Прямий комп'ютер	Amd	C++ AMP
Підтримувані технології	OpenGL, Direct3D,Пряма сумісність з openCL	OpenGL, Відкритий доступ		Opencl	OpenCL, OpenGL, Пряма комп'ют
Наявність зручних інструментів розробки	Так	Так	Ні	Так	Так
Коментарі				Розширення OpenCL	Розширення DirectCompute

Таблиця 4 Короткий огляд архітектур паралельних обчислень на ГПУ

Аналізуючи джерела 27-30 і таблицю на їх основі, можна зробити висновок, що архітектура CUDA є найбільш універсальною і зручною.

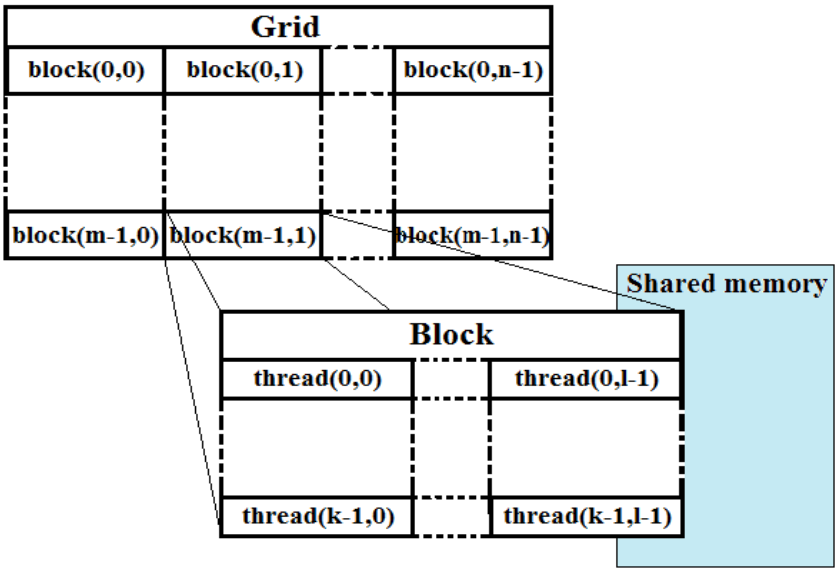
Огляд архітектури CUDA

Nvidia була однією з перших, хто представив цю технологію на своїй відеокарті восьмого покоління, G80 (GeForce 8800 G GTX,2006) і як і раніше підтримує і розробляє її на своїх продуктах.

Nvidia не розкриває точні цифри продажів для своїх пристроїв, але можна стверджувати, що при нинішньому розвитку CUDA ігрової індустрії продажі відеокарт GeForce знаходяться в сотні мільйонів. Nvidia quatro використовується в багатьох комп'ютерних графічних компаніях, Nvidia Tesla використовується в обчислювальних центрах багатьох інститутів і науково-дослідних інститутів.

Створення, керування та знищення потоку на графічному процесорі вимагає набагато менше часу та ресурсів, ніж на процесорі, завдяки тому, що всі налаштування GPU пасма визначені CPU заздалегідь.

Пасма в CUDA організовані за наступною схемою (рис. 8): основою є одновимірна або grid двовимірна сітка, що складається з одновимірних, двовимірних або об'ємних блоків, що складаються з ниток. .



Малюнок 7 Ієрархія в CUDA

Ця ієрархія спрощує паралельну багатовизначну обробку даних. Крім того, потоки окремого блоку можна синхронізувати з усіма іншими потоками блоку (синхронізація бар'єрів - немає потоку, досягнувши бар'єру синхронізації, не

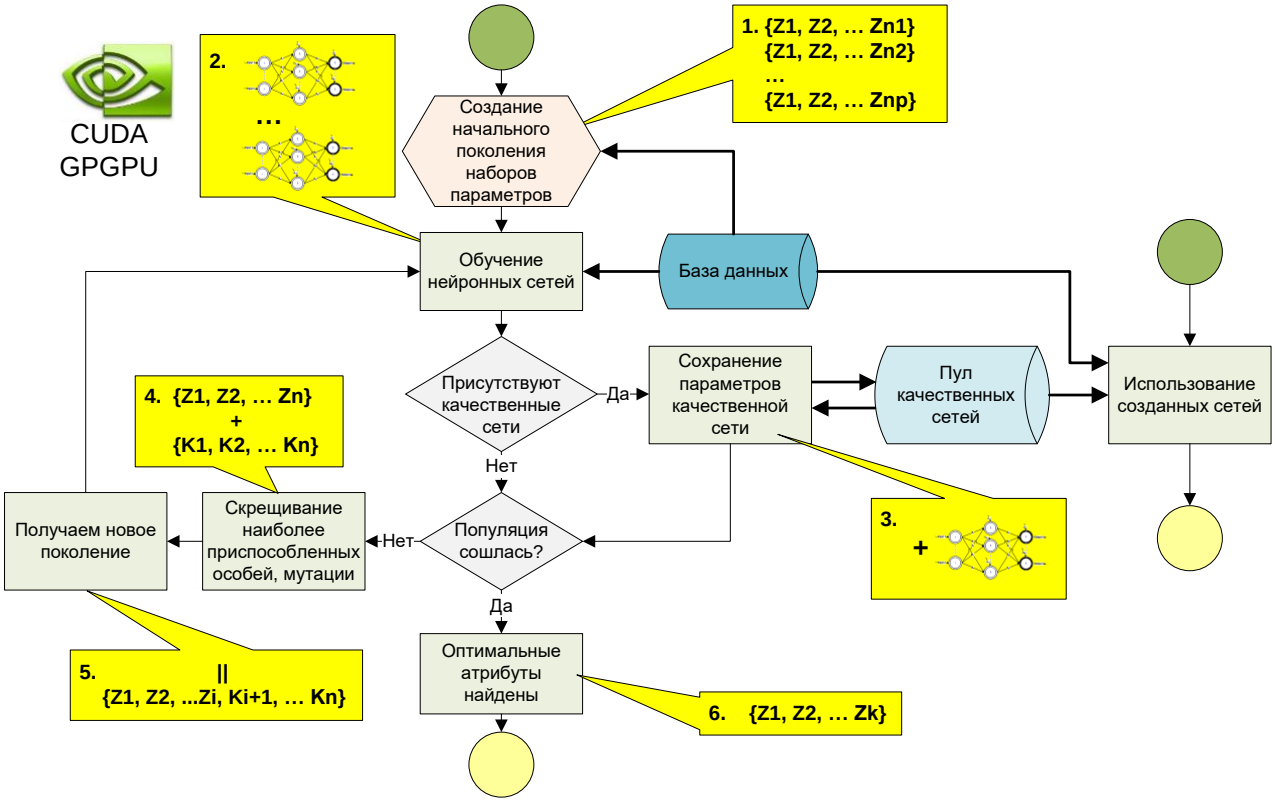
буде продовжуватися, поки не буде досягнуто всіх) і використовувати спільну пам'ять блоку (спільна пам'ять), щоб прийняти в цілому від 1 до 4 барів, в той час як загальна пам'ять графічного GPU процесора (глобальна пам'ять) займає від 100 до 1000 барів. Тому, коли ви працюєте більш-менш активно з даними рекомендується записати їх у спільну пам'ять перед роботою з даними, а після блоку записати результат назад обратно у глобальну пам'ять ГПУ.

Таким чином, використовуючи пакет процесора та графічного процесора, ви можете досягти простого ПК з раніше невидимою продуктивністю.

3 РЕАЛІЗАЦІЯ ПРОГНОЗУВАННЯ ЧАСОВОГО РЯДУ

3.1 Функціональна схема системи прогнозування часових рядів та деталі реалізації

Коли ми поєднуємо генетичний алгоритм і інтервальну нейронну мережу, ми отримуємо універсальну систему прогнозування, яка працює прийнятний час. Функціональна схема системи, що створюється, представлена на рисунку 9.



Малюнок 8 Функціональна схема оболонки системи, що розробляється

Перед запуском алгоритмів пошуку оптимальної нейронної мережі для прогнозування взаємозв'язковості поточного параметра користувач може встановити систему наступним чином:

- Ім'я та адреса файлу CSV, що містить часовий ряд для навчання мережі
- Ім'я та адреса файлу CSV,що містить часовий ряд, на основі якого передбачається прогнозування.

- Шлях до каталогу полягає у збереженні якісних мереж.
- Атрибут і прогнозоване число
- Діапазон атрибутів, які використовуються для прогнозування
- Діапазон вікон часу для атрибутів, на яких буде зроблено прогноз (залежно від того, скільки послідовних значень часових рядів для створення прогнозу)
- Розмір вікна часу прогнозованого значення (скільки послідовних значень потрібно передбачити)
- Розмір проміжку часу між вікном атрибута та прогнозованим вікном (може бути від'ємним, наприклад, для оцінки однакової величини за їх продуктивністю,які знаходяться в одному проміжку часу з орієнтовним значенням)
- Максимальні та мінімальні значення досліджуваних значень (для нормалізації)
- Чисельність населення для генетичного алгоритму
- Кількість внутрішніх нейронів для нейронної мережі
- Кількість навчальних циклів для нейронних мереж
- Кількість нейронних мереж, навчених за одним набором параметрів
- Максимальна кількість ітерацій

Алгоритм оболонки системи:

1. Першим кроком алгоритму пошуку, оптимальним для прогнозування нейронної мережі, є створення початкових поколінь. Для кожного значення діапазону використовуваних атрибутів і діапазону вікон часу формується встановлене користувачем число (розмір популяції) випадкових наборів параметрів, на основі якого буде базуватися прогноз.
2. На наступному кроці для кожного набору параметрів система тренує багато нейронних мереж, які передбачають суму, яку ви шукаєте. Навчання відбувається паралельно з використанням архітектури CUDA

для паралелізації обчислень на графічних процесорах, що значно скорочує час алгоритму.

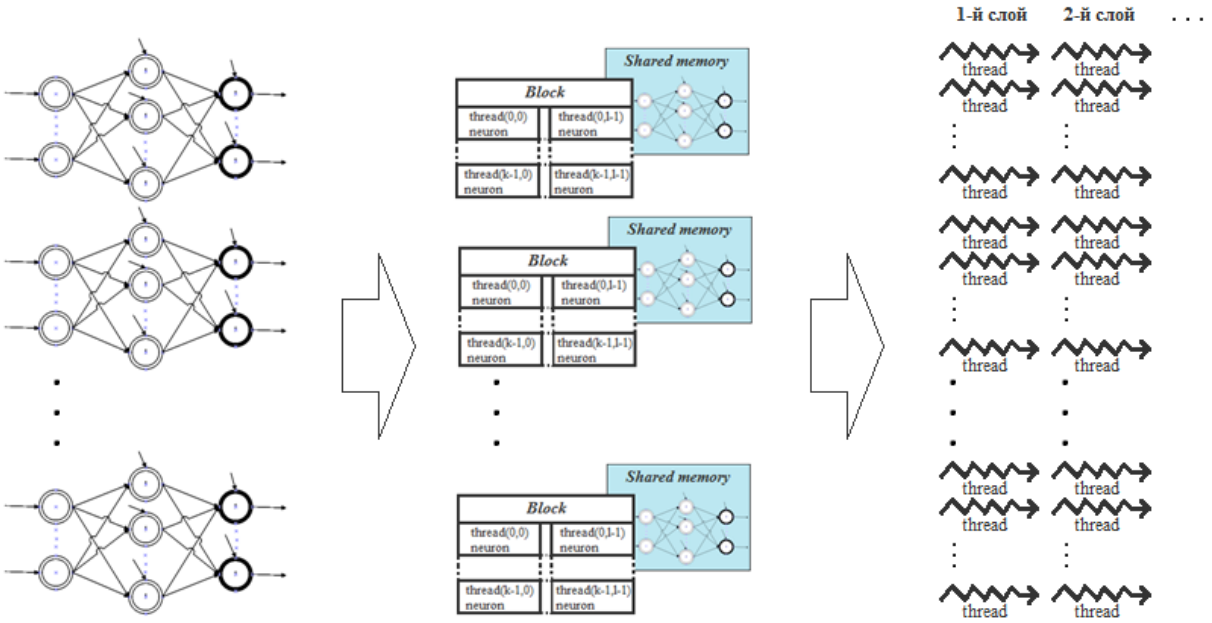
3. Якщо серед навчених мереж є мережа, яка працює зе меншою похибкою, ніж решта набору, параметри такої мережі повинні зберігатися в пулі якісних мереж. Як тільки перша мережа з'явиться в басейні, паралельно з алгоритмом пошуку оптимальних параметрів нейронної мережі може працювати алгоритм прогнозування, вибираючи з басейну мережу з найменшою похибкою.
4. Якщо населення не сходиться або максимальна кількість ітерацій не перевищена, кожне покоління визначає набір кращих осіб з найменшою похибкою прогнозування для взаємозв'язку.
5. На основі обраних кращих індивідуумів генеруються нові покоління шляхом перетину і мутацій. Алгоритм продовжується з пункту 2.
6. Програма виходить з циклу або в разі перевищення максимальної кількості ітерацій, або в разі зближення населення. Якщо населення сходилося, то на виході ми отримуємо оптимальний набір атрибутів, а нейронна мережа на його основі вже знаходиться в пулі якісних мереж.

Як видно зі схеми, однією з ключових переваг системи є можливість прогнозуватие вже на ранніх стадіях генетичного алгоритму (паралельно з його роботою).

Другою і головною перевагою оболонки є паралелізм найбільш трудомісткої процедури даного рішення - навчання багатьох нейронних мереж. Використовуючи архітектуру CUDA, швидкість цього алгоритму значно вища, ніж у аналогічної реалізації тільки за участю процесора.

З огляду на те, що кожен нейрон в мережі може розраховуватися незалежно від решти нейронів в своєму шарі, а також від того, що кожна окрема нейронна мережа активно взаємодіє з власними парами лічильників (синаптичних ваг), розроблений і впроваджений наступний метод паралелізатора: крім одночасного навчання нейронних мереж одного покоління,

кожна мережа паралелізується нейронами в межах окремого блоку. Ілюстрація цього методу показана на рисунку 10.



Малюнок 9 Ілюстрація паралельних обчислень

3.2 Моделювання процесу прогнозування часних рядів

Для перевірки якості оболонки системи було прийнято рішення CPU. провести серію прогнозування часових рядів з складними неочевидних залежностей.

Для порівняння точності методу було проведено 20 тестів для прогнозування різних показників (курси валют, ціни акцій провідних IT-компаній, STATISTICA

Інструмент прогнозування	Середеня похибка у відсотках	Середній час
Оболонка SPPR	2	0,58 сек.
СТАТИСТИКА	3	0,6 сек.
СТАТИСТИКА	1.94	За 7 хвилин

Таблиця 5 Порівняльні характеристики інструментів прогнозування

Також були проведені випробування швидкості паралельного і послідовного виконання розробленої оболонки, результати представлені в таблиці 6.

Тип обчислювальної техніки	Кількість мереж	Кількість нейронів	Кількість ітерацій	Кількість навчальних циклів	Час роботи
Послідовне	1	27	10	1 000	0,51 сек.
	2	54	10	1 000	1,07 сек.

Тип обчислювальної техніки	Кількість мереж	Кількість нейронів	Кількість ітерацій	Кількість навчальних циклів	Час роботи
	3	81	10	1 000	1,66 сек.
Паралельно	1	27	10	1 000	0,47 сек.
	2	54	10	1 000	0,53 сек.
	3	81	10	1 000	0,58 сек.

Таблиця 6 Порівняльні характеристики швидкості послідовної та паралельної реалізації

Було проведено дві серії експериментів з перевірки здатності розробленої оболонки виявляти складні неочевидні залежності.

Перша серія експериментів досліджувала якість прогнозу по долару США(ДОЛ) в залежності від вхідної інформації.

Таблиця 6 містить інформацію про запропоновані часовий ряд і результати, отримані від запропонованих даних.

Ряди біля входу в мушлі	Діапазон атрибутів, що використовуються	Рядки, вибрані оболонкою	Середня помилка abs.
Долар США	1	Долар США	5. 3.3.3.3.33
Курси: USD USD; MXN; євро;; AUD; NZD; швейцарських	3-6	:ДОЛАРИ США USD; євро; AUD;;	3. 06 .0

франків; руб.;; фунтів стерлінгів; CNY; (AED); (у 200 за все; (у 200 за все		фунтів стерлінгів; CNY; (у 200 за все	
Курс: USD USD; MXN; євро;; AUD; NZD; швейцарських франків; руб.;; фунтів стерлінгів; CNY; (AED); (у 200 за все; (у 200 за все Ціни на метал: золото, срібло, платина	3-10	Куры: USD USD; євро; AUD;; NZD; швейцарських франків;; фунтів стерлінгів;; (у 200 за все Ціни на метал: золото, платина	2. 22.2.2.2.2
Курс: USD USD; MXN; євро;; AUD; NZD; швейцарських франків; руб.;; фунтів стерлінгів; CNY; (AED); (у 200 за все; (у 200 за все Основні індекси: DJIA; FTSE; CAC 40;; DAX; Nikkei; (у 200 за все	3-10	Куры: USD USD; євро;; AUD;; фунтів стерлінгів;; CNY; (у 200 за все Основне индексы: DJIA; FTSE; DAX; Nikkei;	2. 78
Курси: USD USD; MXN; євро;; AUD; NZD; швейцарських франків; руб.;; фунтів стерлінгів; CNY;	3-10	Куры: USD USD; євро; AUD;; NZD; фунтів стерлінгів Основні індкекс:	1. 96 1000000009

(AED); (у 200 за все; (у 200 за все Основне индексы: DJIA; FTSE; CAC 40; DAX; Nikkei; (у 200 за все Ціни на метал: золото, срібло, платина		DJIA; FTSE; Nikkei; Ціни на метал: золото	

Таблиця 7 Перша серія експериментів

Аналізуючи результати випробувань, можна прийти до висновку, що розроблена оболонка системи підтримки прийняття рішень справляється із завданням виявлення складних залежностей і здатна отримати змістовні результати.

ВИСНОВОК

Метою цієї роботи була розробка оболонки системи підтримки прийняття рішень, яка дозволила б прогнозувати часові ряди в системах зі складнимив заємними залежностями, дозволяючи їм працювати з набором даних з інтервалом, а також мала більш високу продуктивність і була реалізована на звичайних обчислювальних інструментах - персональних комп'ютерах.

Для досягнення цієї мети перед роботою було поставлено ряд завдань, які були успішно вирішені.

При вирішенні проблеми зміни алгоритму зворотного розподілу похибки для навчання інтервальної нейронної мережі, що дозволяють працювати з невизначеними даними, я розглянув роботи і на їх основі розробили спосіб навчання інтервальних нейронних мереж загального роду.

Для вирішення проблеми модифікації генетичного алгоритму і визначення оптимальної структури інтервалної нейронної мережі, а також визначення необхідного обсягу інформації про ці параметри для оперативного і точного рішення, робота вивчила основні принципи застосування генетичних алгоритмів, а також деталізувала метод взаємодії генетичних алгоритмів спільно з нейронними мережами.

Щоб скоротити час, який знадобився для аналізу керованої системи та розробки рішення управління, запропоновані алгоритми були реалізовані за допомогою паралельної обчислювальної архітектури на процесорах графічного процесора. Вивчено принципи використання графічних графічних завдань та розроблено модель для електронної директивної паралелізації купи еволюційних алгоритмів з використанням архітектури CUDA.

Реалізована оболонка системи підтримки прийняття рішень була на реальних прогнозуваннях зі складними залежностями, щоб порівняти її продуктивність з існуючими системами і методами.

Виходячи з аналізу використовуваних методів і випробувань фінальної системи, можна прийти до наступних висновків:

Ця оболонка систем є реалізацією принципово нової моделі SPPR, що дозволяє працювати з неточними, випадковими або стандартними даними інтервалу.

Розроблена оболонка успішно справляється зі своїм завданням при роботі з даними складної структури і складних взаємних залежностей. Це дозволяє нейронним мережам, які розглядають до 30 істотних вхідних параметрів, викладатися без значної втрати ефективності і точності.

Бажано застосовувати розроблену оболонку в досить складних завданнях прогнозування, таких як прогнозування щоденних валютних змін, прогнозування соціальних процесів тощо для кого важко або неможливо визначити відповідний статистичний метод.

Також використання цієї оболонки доцільно при роботі з даними інтервалу з неточними значеннями з помилкою, випадковими значеннями, швидко змінюючи безперервні значення поза контролем більшості методів. При застосуванні методів прогнозування, які базують свої прогнози на основі статистики, необхідно мати на увазі, що при появі принципово нових факторів або раніше нерозглянутих показників прогноз може бути неточним або взагалі невірним

Наукова новизна цієї роботи є запропонованим принципово новим геристиковим методом пошуку оптимальної структури і параметрів інтервальної нейронної мережі для прогнозування часових рядів, заснованих на взаємодії нейронних мереж і генетичних алгоритмів.

Практичне значення роботи полягає в розробці програмного продукту,який впроваджує методи і підвищує якість навчання нейронної мережі і, як наслідок, якість знань і точність прогнозування, набуті цими мережами. Також наукову новинку можна віднести до збільшення швидкості тренувальних наборів нейронних мереж.

Таким чином, завдання повністю вирішені, досягнута мета роботи - розроблено і впроваджено оболонку системи підтримки прийняття рішень на основі генетичного алгоритму, нейронних мереж і мас-паралельних обчислень на графічних процесорах.

Напрямками подальших досліджень можуть бути:

- Удосконалення прогнозування шляхом розробки нових методів аналізу даних з використанням нейронних мереж.
- Збільшення швидкості навчання нейронних мережза умови розвитку паралельних обчислювальних технологій на графічних процесорах.

СПИСОК ЛІТЕРАТУРИ

1. Саати, Т. Л. Принятие решений. Метод анализа иерархий / Т. Л. Саати. : Радио и связь, 1993. - 278с.

2. Druzdzel, M. J. Decision Support Systems / M. J. Druzdzel, R. R. Flynn // Encyclopedia of Library and Information Science. Second Edition. - New York: Marcel Dekker, Inc., 2020. -15p.

3. Alter, S. L. Decision Support Systems: Current Practice and Continuing Challenge / S. L. Alter. - MA: Addison-Wesley, 1980. - 316p.

4. Haettenschwiler, P. Neues anwenderfreundliches Konzept der Entscheidungs-unterstutzung / P. Haettenschwiler // Gutes Entscheiden in Wirtschaft, Politik und Gesellschaft. - Zurich: Hochschulverlag AG, 1999. - P. 189-208.

5. Power, D. J., A Brief History of Decision Support Systems. DSSResources.COM / D. J. Power. - version 4.0, March 10, 2007 -.- Режим доступа : http://DSSResources.COM/history/dsshistory.html

6. Holsapple, C. W. Decision Support Systems: A Knowledge-based Approach / C. W. Holsapple, A. B. Whinston. - Minneapolis: West Publishing Co., 2019. -713p.

7. Golden, B. Decision Insight Systems: A Critical Evaluation / Golden B., Hevner A., Power D.J. // Computers and Operations Research, v. 13. - N2/3. - 1986. - P. 287-300.

8. Объединенный пресс релиз IDC и EMC. Digital Universe study "Extracting Value from Chaos" http://www.emc.com/about/news/press/2011/20110628-01.htm, http://www.emc.com/collateral/demos/microsites/emc-digital-universe-2011/index.htm.

9. Черняк, Л. Большие Данные — новая теория и практика / Л. Черняк // Открытые системы. № 10. - М.: Открытые системы, 2011. .http://www.osp.ru/os/2011/10/13010990/, свободный.

10. Simon, H. A. The Structure of Ill-structured Problems. In Developments in Design Methodology / H. A. Simon // Cross N. (ed.). - UK Chichester: J. Wiley & Sons, 1984 - P. 317-327.

11. Шмойлова, Р. А. Общая теория статистики: Учебник, / Р. А. Шмойлова. - М.: Финансы и статистика, 2002. - 415 с.

12. Manning, C. Introduction to Information Retrieval / C. Manning, P. Raghavan, H. Schutze. - Cambridge University Press, 2008. - 544 p.

12. Николенко, С. Конспект лекции по деревьям принятия решений / С. Николенко. - 2020.: http://logic.pdmi.ras.ru/~sergey/teaching/ml/notes-01-dectrees.pdf,

13. Духанов, А. В. Имитационное моделирование сложных систем, Курс лекций / А. В. Духанов, О. Н. Медведева. - Владимир: Изд-во Владим. гос. ун-та, 2016. - 115 с.

15. Авдеев, З. К. Когнитивное моделирование для решения задач управления слабоструктурированными системами (ситуациями) / З. К. Авдеев, С. В. Коврига, Д. И. Макаренко // Управление большими системами. Выпуск 16. - М.: ИПУ РАН, 2007. - С. 26-39.

16. Логунова, Е. А. Математические модели систем поддержки принятия решений / Е. А Логунова. - Смоленск, 2012. -.- Режим доступа : http://sibac.info/index.php/2009-07-01-10-21-16/3423-2012-07-29-14-43-08, свободный. – Загл. с экрана.

17. Ивченко, Г. И. Введение в математическую статистику: Учебник / Г.И., Ивченко Ю.И. Медведев. - М.: Издательство ЛКИ, 2010. - 600с.

18. Магнус, Я. Р. Эконометрика. Начальный курс: Учеб. 8-е изд., испр. / Я. Р. Магнус, П. К. Катышев, А. А. Пересецкий. - М.: Дело, 2007. - 400с.

19. Горбань, А. Н. Обобщенная аппроксимационная теорема и вычислительные возможности нейронных сетей / А. Н. Горбань // Сибирский журнал вычислительной математики, Т.1, № 1. - Издательство СО РАН, 1998. - С. 12-24.

20. Хайкин, С. Нейронные сети: полный курс, 2-е изд., испр. / С. Хайкин. - М.: Издательский дом "Вильямс", 2006. - 1104с.

21. Царегородцев, В. Г. Преимущества и достоинства нейронных сетей / В. Г. Царегородцев. -.- Режим доступа : http://www.neuropro.ru/neu3.shtml, свободный. – Загл. с экрана.

22. Ishibuchi, H. An architecture of neural networks with interval weights and its application to fuzzy regression analysis / H. Ishibuchi, H. Tanaka // Fuzzy Sets and Systems, 57. - North-Holland, 1993. - P. 27-39.

23. Ishibuchi, H. An extension of the BP-algorithm to interval input vectors / H. Ishibuchi, H. Tanaka // Proc. IEEE Int. Joint Conf. on Neural Networks, Vol.2. - Singapore, 1991 - P. 1588-1593.

24. Емельянов, В. В. Теория и практика эволюционного моделирования / В. В. Емельянов, В. В. Курейчик, В. М. Курейчик. - М.: Физматлит, 2003. - 432 с.

25. Рутковская, Д. Нейронные сети, генетические алгоритмы и нечеткие системы / Д. Рутковская, М. Пилиньский, Л. Рутковский. - М.: Горячая линия - Телеком, 2006. - 452 с.

26. Nvidia. Tesla gpu accelerators for servers. -.- Режим доступа : http://www.nvidia.com/object/tesla-servers.html, свободный. – Загл. с экрана.

27. Nvidia. CUDA Toolkit Documentation. -.- Режим доступа : http://www.nvidia.com/object/tesla-servers.html, свободный. – Загл. с экрана.

28. Microsoft. C++ Accelerated Massive Parallelism (C++ AMP). -.- Режим доступа : http://blogs.msdn.com/cfs-file.ashx/__key/communityserver-components-postattachments/00-10-29-86-29/cppAMPv6_2D00_gen.pdf/, свободный. – Загл. с экрана.

29. AMD. ATI STREAM programming guide. -.- Режим доступа : http://www.amddevcentral.com/gpu_assets/ATI_Stream_SDK_CAL_Programming_Guide_v2.0.pdf, свободный. – Загл. с экрана.

30. AMD. Accelerated Parallel Processing OpenCL Programming Guide. -.-: http://developer.amd.com/wordpress/media/2012/10/AMD_Accelerated_Parallel_Processing_OpenCL_Programming_Guide.pdf, свободный. – Загл. с экрана.

31. Nvidia. What is CUDA. -.- Режим доступа : http://docs.nvidia.com/cuda/index.html, свободный. – Загл. с экрана.

28. Борезков, А. В. Основы работы с технологией CUDA / А. В. Борезков, А. А. Харламов. - М.: ДМК Пресс, 2010. - 232 с.

29. Бухаров, О. Е. Задача прогнозирования временных рядов. Подход к решению с использованием эволюционных алгоритмов / О. Е. Бухаров, А. А. Мизикин // Научно-техническая конференция студентов, аспирантов и молодых специалистов МИЭМ НИУ ВШЭ. Тезисы докладов. - М.: МИЭМ НИУ ВШЭ, 2013. - С. 73.

30. Бухаров, О. Е. Разработка оболочки системы прогнозирования временных рядов с использованием эволюционных алгоритмов / О. Е. Бухаров, Д. П. Боголюбов // Новые информационные технологии: Сборник трудов XVI Всероссийской научно-технической конференции. - М.: МГУПИ, 2013. - С. 63-68.

31. Бухаров, О. Е. Разработка оболочки системы поддержки принятия решений с использованием эволюционных алгоритмов / О. Е. Бухаров, А. А. Мизикин, Д. П. Боголюбов // Промышленные АСУ и Контроллеры. - М.: НАУЧТЕХЛИТИЗДАТ - в печати.

32. Круглински, Д. Дж. Программирование на Microsoft Visual C++ 6.0 / Д.Дж. Круглински, С. Уингоу, Дж. Шеферд. - Санкт-Петербург: Питер, 2003. - 819 с.

33. Сандерс, Дж. Технология CUDA в примерах: введение в программирование графических процессоров / Дж. Сандерс, Э. Кэндрот. - М.: ДМК Пресс, 2011. - 232 с.

Додаток А. Елементи діпломного коду СППР

Ядро прямого розподілу сигналу в нейронній мережі

```
__global__ кормиForward(float* gpuWih, float* gpuЯкий, float * gpuInputs, float *
gpuActual, int INPUT_NEURONS, int HIDDEN_NEURONS, int
OUTPUT_NEURONS, cudaStream_t потік)
{
  int thx=threadIdx.x; Вводу
  int твій = threadIdx.y; Приховані
  int thabs=HIDDEN_NEURONS*thx+твоя;
  int сховався, вийшов;

        зовнішня __shared__ спільна кома[];
(INPUT_NEURONS+HIDDEN_NEURONS+OUTPUT_NEURONS)*2+(INPUT_N
EURONS+1)*HIDDEN_NEURONS+(HIDDEN_NEURONS+1)*OUTPUT_NEUR
ONS+max_NEURONS*HIDDEN_NEURONS*2
        float * входи = sharedMem;
        float *hidden = &sharedMem[INPUT_NEURONS*2];
        float *actual = &hidden[HIDDEN_NEURONS*2];
        float *wih = &actual[OUTPUT_NEURONS*2];
        float *who = &wih[(INPUT_NEURONS+1)*HIDDEN_NEURONS];
        float *subsum = &who[(HIDDEN_NEURONS+1)*OUTPUT_NEURONS];

  2000000000000000000000000000000000000000000000000
  for(int i=thabs;
i<(INPUT_NEURONS*2+HIDDEN_NEURONS+(HIDDEN_NEURONS+1)*OUTP
UT_NEURONS); i+=INPUT_NEURONS*HIDDEN_NEURONS)
  {
```

```
якщо(i<INPUT_NEURONS*2)
        inputs[i]=gpuInputs[i];
інші if(i<INPUT_NEURONS*2+HIDDEN_NEURONS)
        wih[HIDDEN_NEURONS*INPUT_NEURONS+i-
INPUT_NEURONS*2]=gpuWih[HIDDEN_NEURONS*INPUT_NEURONS+i-
INPUT_NEURONS*2];
ЩЕ
        хто[i-(INPUT_NEURONS*2+HIDDEN_NEURONS)]=gpuЯкий[i-
(INPUT_NEURONS*2+HIDDEN_NEURONS)];
 }

__syncthreads();
 /* Обчислити вхід до прихованого шару */

mult(&входи[thx*2],
wih[thabs],&subsum[(thy*INPUT_NEURONS+thx)*2]);
__syncthreads();
int s = 2;
у той час як(s<INPUT_NEURONS)
        s<<=1;
  у той час як (s>1)
{
        s>>=1;
        if(thx & s && thx+s < INPUT_NEURONS)
        {
                subsum[(твоя*INPUT_NEURONS+thx)*2] +=
subsum[(твоя*INPUT_NEURONS+thx+s)*2];
                subsum[(твій*INPUT_NEURONS+thx)*2+1] +=
subsum[(thy*INPUT_NEURONS+thx+s)*2+1];
        }
```

```
            __syncthreads();
    }

    /* Додати в ухил */
        if(thx==0)
        {
            hidden[thy*2] = sigmoid(subsum[thy*INPUT_NEURONS*2]+
wih[INPUT_NEURONS*HIDDEN_NEURONS+твоя] );
            hidden[thy*2+1] = sigmoid(subsum[thy*INPUT_NEURONS*2+1]+
wih[INPUT_NEURONS*HIDDEN_NEURONS+thy] );
        }

        __syncthreads();

  /* Обчислити прихований до виводу шар */
  для (out = thabs ; out < OUTPUT_NEURONS ;
out+=INPUT_NEURONS*HIDDEN_NEURONS)
  {
    for(hid=0; hid<HIDDEN_NEURONS; hid++)
        {
            mult(&hidden[hid*2],
який[OUTPUT_NEURONS*hid+out],&subsum[(out*HIDDEN_NEURONS+hid)*2
]);

            if(hid > 0)
            {
                subsum[(out*HIDDEN_NEURONS)*2] +=
subsum[(out*HIDDEN_NEURONS+hid)*2];
                subsum[(out*HIDDEN_NEURONS)*2+1] +=
subsum[(out*HIDDEN_NEURONS+hid)*2+1];
```

```
            }
    }

        /* Додати в ухил */
        фактичний[out*2] = sigmoid(subsum[out*HIDDEN_NEURONS*2]+,
який[HIDDEN_NEURONS*OUTPUT_NEURONS+out] );
        фактичний[out*2+1] = sigmoid(subsum[out*HIDDEN_NEURONS*2+1]+,
який[HIDDEN_NEURONS*OUTPUT_NEURONS+out] );

    }

    __syncthreads();
    for(int i=thabs; i<OUTPUT_NEURONS*2;
i+=INPUT_NEURONS*HIDDEN_NEURONS)
    {
        gpuActual[i]=фактичний[i];
    }
}
```

Ядро мережевого навчання на основі повторного розповсюдження помилок

```
__global__ backPropagate(float* gpuWih, float * gpuDWih, float * gpu, float *
gpuDЯкий, float * gpuInputs, float * gpuActual, float* gpuTarget, int
INPUT_NEURONS, int HIDDEN_NEURONS, int OUTPUT_NEURONS) //
<<<1,dim3(INPUT_,HIDDEN_),.
(INPUT_NEURONS+HIDDEN_NEURONS+OUTPUT_NEURONS*2+(INPUT_N
EURONS+1)*HIDDEN_NEURONS+(HIDDEN_NEURONS+1)*OUTPUT_NEUR
ONS+HIDDEN_NEURONS+OUTPUT_NEURONS+max_NEURONS*HIDDEN_N
EURONS)*2*sizeof(float)>>>
```

```
{
  int out,
  int thx=threadIdx.x; input
  int your= threadIdx.y; Приховані
  int thabs=HIDDEN_NEURONS*thx+your;

  зовнішня __shared__ спільна кома[];
  float * inputs= sharedMem;
  float *hidden = &sharedMem[INPUT_NEURONS*2];
  float *actual = &hidden[HIDDEN_NEURONS*2];
  float *target = &actual[OUTPUT_NEURONS*2];
  float *wih = &target[OUTPUT_NEURONS*2];
  float *dwih = &wih[(INPUT_NEURONS+1)*HIDDEN_NEURONS];
  float *who = &dwih[(INPUT_NEURONS+1)*HIDDEN_NEURONS];
  float *dwho = &who[(HIDDEN_NEURONS+1)*OUTPUT_NEURONS];
  float *subsum = &dwho[(HIDDEN_NEURONS+1)*OUTPUT_NEURONS];

  for(int i=thabs;
i<(INPUT_NEURONS+OUTPUT_NEURONS+(HIDDEN_NEURONS+1)*OUTPU
T_NEURONS+HIDDEN_NEURONS)*2;
i+=INPUT_NEURONS*HIDDEN_NEURONS)
  {
        якщо(i<INPUT_NEURONS*2)
        {
              inputs[i]=gpuInputs[i];
        }
        інші if(i<(INPUT_NEURONS+OUTPUT_NEURONS)*2)
        {
              [i-INPUT_NEURONS*2]=gpuTarget[i-INPUT_NEURONS*2];
        }
```

```
        інші
if(i<(INPUT_NEURONS+OUTPUT_NEURONS)*2+(HIDDEN_NEURONS+1)*O
UTPUT_NEURONS)
        {
                хто[i-(INPUT_NEURONS+OUTPUT_NEURONS)*2]=gpuЯкий[i-
(INPUT_NEURONS+OUTPUT_NEURONS)*2];
        }
        інші
if(i<(INPUT_NEURONS+OUTPUT_NEURONS+(HIDDEN_NEURONS+1)*OUT
PUT_NEURONS)*2)
        {
                dwho[i-
(INPUT_NEURONS+OUTPUT_NEURONS)*2+(HIDDEN_NEURONS+1)*OUTP
UT_NEURONS)]=gpuDWho[i-
(INPUT_NEURONS+OUTPUT_NEURONS)*2+(HIDDEN_NEURONS+1)*OUTP
UT_NEURONS)];
        }
        інші
if(i<((INPUT_NEURONS+OUTPUT_NEURONS+(HIDDEN_NEURONS+1)*OUT
PUT_NEURONS)*2+HIDDEN_NEURONS))
        {
                wih[i-
(INPUT_NEURONS+OUTPUT_NEURONS+(HIDDEN_NEURONS+1)*OUTPUT
_NEURONS)*2+INPUT_NEURONS*HIDDEN_NEURONS]=gpuWih[i-
(INPUT_NEURONS+OUTPUT_NEURONS+(HIDDEN_NEURONS+1)*OUTPUT
_NEURONS)*2+INPUT_NEURONS*HIDDEN_NEURONS];
        }
        ІЩе
        {
```

```
                dwih[i-
(INPUT_NEURONS+OUTPUT_NEURONS+(HIDDEN_NEURONS+1)*OUTPUT
_NEURONS)*2-
HIDDEN_NEURONS+INPUT_NEURONS*HIDDEN_NEURONS]=gpuDWih[i-
(INPUT_NEURONS+OUTPUT_NEURONS+(HIDDEN_NEURONS+1)*OUTPUT
_NEURONS)*2-
HIDDEN_NEURONS+INPUT_NEURONS*HIDDEN_NEURONS];
        }
  }
  2000000000000000000000000000000000000000000000000
  dwih[thabs]=gpuDWih[thabs];

  __syncthreads();



                подача Вперед

  /* Обчислити вхід до прихованого шару */

        mult(&входи[thx*2],
wih[thabs],&subsum[(thy*INPUT_NEURONS+thx)*2]);
        __syncthreads();
        int s = 2;
        у той час як(s<INPUT_NEURONS)
                s<<=1;
    у той час як (s>1)
        {
                s>>=1;
                if(thx & s && thx+s < INPUT_NEURONS)
                {
```

```
                    subsum[(твоя*INPUT_NEURONS+thx)*2] +=
subsum[(твоя*INPUT_NEURONS+thx+s)*2];
                    subsum[(твій*INPUT_NEURONS+thx)*2+1] +=
subsum[(thy*INPUT_NEURONS+thx+s)*2+1];
            }
            __syncthreads();
   }

   /* Додати в ухил */
        if(thx==0)
        {
            hidden[thy*2] = sigmoid(subsum[thy*INPUT_NEURONS*2]+
wih[INPUT_NEURONS*HIDDEN_NEURONS+твоя] );
            hidden[thy*2+1] = sigmoid(subsum[thy*INPUT_NEURONS*2+1]+
wih[INPUT_NEURONS*HIDDEN_NEURONS+thy] );
        }

        __syncthreads();

  /* Обчислити прихований до виводу шар */
  для (out = thabs ; out < OUTPUT_NEURONS ;
out+=INPUT_NEURONS*HIDDEN_NEURONS)
  {
    for(hid=0; hid<HIDDEN_NEURONS; hid++)
        {
            mult(&hidden[hid*2],
який[OUTPUT_NEURONS*hid+out],&subsum[(out*HIDDEN_NEURONS+hid)*2
]);

            if(hid > 0)
```

```
            {
                subsum[(out*HIDDEN_NEURONS)*2] +=
subsum[(out*HIDDEN_NEURONS+hid)*2];
                subsum[(out*HIDDEN_NEURONS)*2+1] +=
subsum[(out*HIDDEN_NEURONS+hid)*2+1];
            }
   }

      /* Додати в ухил */
      фактичний[out*2] = sigmoid(subsum[out*HIDDEN_NEURONS*2]+,
який[HIDDEN_NEURONS*OUTPUT_NEURONS+out] );
      фактичний[out*2+1] = sigmoid(subsum[out*HIDDEN_NEURONS*2+1]+,
який[HIDDEN_NEURONS*OUTPUT_NEURONS+out] );

  }

  __syncthreads();



            зворотний з'єд.

  додатковий розрахунок

  float dE, dEe;
  float *b= &dwho[(HIDDEN_NEURONS+1)*OUTPUT_NEURONS]; новий
поплавок[OUTPUT_NEURONS*2]
  float *dh= &b[OUTPUT_NEURONS*2]; новий
поплавок[HIDDEN_NEURONS*2]
```

```
 for(out = thabs ; out < OUTPUT_NEURONS ;
out+=INPUT_NEURONS*HIDDEN_NEURONS)
 {
        b[out*2]=-(ціль[out*2] - фактична[out*2])*actual[out*2]*(1-actual[out*2]);
        b[out*2+1]=-(ціль[out*2+1] - фактична[out*2+1])*actual[out*2+1]*(1-
actual[out*2+1]);
 }

 __syncthreads();

 if(thx==0)
 {
  float s1,s2,s3,s4;
      s1=0;
      s2=0;
      s3=0;
      s4=0;
      for(out = 0 ; out < OUTPUT_NEURONS ; out++)
      {
            якщо(хто[OUTPUT_NEURONS *твій+вихід]>=0)
            {
                  s1+=b[out*2]*who[OUTPUT_NEURONS*thy+out];
                  s3+=b[out*2+1]*who[OUTPUT_NEURONS*thy+out];
            }
            ІЩе
            {
                  s2+=b[out*2+1]*who[OUTPUT_NEURONS*thy+out];
                  s4+=b[out*2]*who[OUTPUT_NEURONS*thy+out];
            }
      }
```

```
        dh[твій*2]=(s1+s2)*прихований[твій*2]*(1-прихований[твій*2]);
        dh[твоя*2+1]=(s3+s4)*прихована[твоя*2+1]*(1-прихована[твоя*2+1]);
  }

  __syncthreads();
  maim обчислення

  for(out = thx ; out < OUTPUT_NEURONS ; out+=INPUT_NEURONS)
  {
                якщо(хто[OUTPUT_NEURONS *твій+вихід]>=0)
                        dE=b[out*2]*hidden[thy*2]+b[out*2+1]*hidden[thy*2+1];
                Ще
                        dE=b[out*2]*hidden[твоя*2+1]+b[out*2+1]*hidden[твоя*2];
                dwho[OUTPUT_NEURONS*твоя+з]*=АЛЬФА;
                dwho[OUTPUT_NEURONS*thy+out]+=LEARN_RATE*(-dE);
                хто[OUTPUT_NEURONS
*твій+вихід]+=dwho[OUTPUT_NEURONS*твоя+з];
          якщо(твій==0)
          {
                  dE=b[out*2]+b[out*2+1];
                  dwho[OUTPUT_NEURONS*HIDDEN_NEURONS+out]*=АЛЬФА;

dwho[OUTPUT_NEURONS*HIDDEN_NEURONS+out]+=LEARN_RATE*(-dE);

хто[OUTPUT_NEURONS*HIDDEN_NEURONS+out]+=dwho[OUTPUT_NEURO
NS*HIDDEN_NEURONS+out];
          }
  }

        якщо(wih[HIDDEN_NEURONS*thx+твоя]>=0)
```

```
            dEe= dh[твої*2]*входи[thx*2]+dh[твій*2+1]*входи[thx*2+1];
        Ще
            dEe= dh[твої*2]*входи[thx*2+1]+dh[твій*2+1]*входи[thx*2];
        dwih[HIDDEN_NEURONS*thx+твоя]*=АЛЬФА;
        dwih[HIDDEN_NEURONS*thx+thy]+=LEARN_RATE*(-dEe);
        200000000000000 HIDDEN_NEURONS
0000000000000000000000000000000000 HIDDEN_NEURONS
00000000000000000000000000000000000000000000
        if(thx==0)
        {
            dEe= dh[твій*2]+dh[твій*2+1];
            dwih[HIDDEN_NEURONS*INPUT_NEURONS+твоя]*=АЛЬФА;

        dwih[HIDDEN_NEURONS*INPUT_NEURONS+твоя]+=LEARN_RATE*( -
dEe);

        wih[HIDDEN_NEURONS*INPUT_NEURONS+твоя]+dwih[HIDDEN_NEU
RONS*INPUT_NEURONS+твоя];
        }

  __syncthreads();
обратний перенос из поділилася  пам'яттю
  gpuWih[thabs]=wih[thabs];
  gpuDWih[thabs]=dwih[thabs];
  for(int i=thabs;
i<(HIDDEN_NEURONS+(HIDDEN_NEURONS+1)*OUTPUT_NEURONS)*2;
i+=INPUT_NEURONS*HIDDEN_NEURONS)
  {
         якщо(i<HIDDEN_NEURONS)
```

```
        gpuWih[INPUT_NEURONS*HIDDEN_NEURONS+i]=wih[INPUT_NEURO
NS*HIDDEN_NEURONS+i];
    інші if(i<HIDDEN_NEURONS*2)
                gpuDWih[(INPUT_NEURONS-
1)*HIDDEN_NEURONS+i]=dwih[(INPUT_NEURONS-
1)*HIDDEN_NEURONS+i];
        інші
if(i<HIDDEN_NEURONS*2+(HIDDEN_NEURONS+1)*OUTPUT_NEURONS)
                gpuWho[i-HIDDEN_NEURONS*2]=хто[i-HIDDEN_NEURONS*2];
        ІЩе
                gpuDWho[i-HIDDEN_NEURONS*2-
(HIDDEN_NEURONS+1)*OUTPUT_NEURONS]=dwho[i-
HIDDEN_NEURONS*2-(HIDDEN_NEURONS+1)*OUTPUT_NEURONS];
  }

}
```