УДК 004.896

Соколенко Д.Г.
Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»

Корнага Я.І.
Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»

СИСТЕМА РОЗПІЗНАВАННЯ ПИСЕМНИХ СИМВОЛІВ ЗА ДОПОМОГОЮ НЕЙРОННОЇ МЕРЕЖІ

Нейронні мережі знаходять все більшу кількість сфер, де вони можуть використовуватись, починаючи від допомоги у захисті інформації на мобільних пристроях і закінчуючи алгоритмами прогнозування будь-яких соціальних, економічних або природних процесів. У статті буде розкрито питання роботи однієї з таких нейронних мереж для розпізнавання рукописних символів. Під час аналізу цього питання буде виявлено, як можна навчити мережу розпізнавати символи та як покращити роботу такого алгоритму за допомогою самонавчання нейронної мережі. Також буде розглянуто питання роботи нейронної мережі за некоректних вхідних даних та які перспективи роботи такої системи у цілому. Загалом стаття має розкрити питання призначення нейронної мережі та доцільність її використання.

Ключові слова: *нейронна мережа, нейрон, машинне навчання, алгоритм, система.*

Постановка проблеми. У людині бувають випадки у житті, коли вона хоче зекономити час на переписування тексту з паперового вигляду на електронний пристрій. Одна з таких ситуацій може відбуватися, коли студент для підготовки до екзамену хоче бачити свій конспект у електронному вигляді, тоді як він записував навчальний матеріал тільки на папері. У такій ситуації може використовуватись нейронна мережа, що може перевести текст із зошита в електронний вигляд.

Аналіз останніх досліджень і публікацій. Останнім часом алгоритми роботи додатка від Google – Google Translate допомагає побачити, наскільки світові компанії намагаються вдосконалювати алгоритми роботи з текстом (і не тільки з ним), оскільки у довгостроковій перспективі такі алгоритмі відіграватимуть значну роль у житті суспільства. Дослідження та публікації ми можемо побачити в офіційному блозі компанії Google – AI Google Blog, де останні статті присвячені розпізнаванню музики або того, що зображено на картинках.

Якщо говорити не тільки про компанію Google, то можна навести одну зі статей з Міжнародної конференції з комп'ютерних наук в Індонезії у 2017 році (ICCSCI 2017), де порушувалося питання роботи нейронних мереж у розпізнаванні символів порівняно з іншими алгоритмами (наприклад, оптичними алгоритмами) розпізнавання тексту, не використовуючи нейромережу. У статті зазначалося, що нейронна мережа у будь-якому разі краще розпізнає текст, ніж будь-які інші системи, які розроблені на цей час.

Постановка завдання. Система розпізнавання писемних символів буде складатися з додатка, який оброблятиме вхідні зображення та оформлюватиме текст в електронному вигляді. Головне завдання нейронної мережі – розпізнавання писемних символів із зображення та виявлення аналогів в електронному вигляді.

Експеримент буде полягати у розробці програми, яка зможе розпізнавати текст із зображення. Під час розробки буде проаналізований алгоритм роботи системи, оброблено інформацію щодо коректності роботи такої системи та знаходження шляхів, як можна покращити роботу нейронної мережі.

Виклад основного матеріалу. Штучна нейронна мережа – це математична модель, яка розроблена за прикладом роботи людського мозку. Основним складником будь-якої нейронної мережі є нейрон. Для штучної нейронної мережі характерна можливість навчання. Це її основна відмінність від роботи звичайних алгоритмів [1].

Для будь-якої нейронної мережі основними складниками є вхідні та вихідні дані. Якщо говорити про вхідні дані (вхідний шар) щодо нашого експерименту, то головним елементом є зображення, на якому присутній текст. Якщо говорити про вихідні дані (вихідний шар), то головним елементом буде електронний текст, який був перенесений з паперового вигляду. Етапом обробки тек-

сту (прихований шар) будуть алгоритми, які мають обробляти вхідні дані та вивести вихідні дані [2].

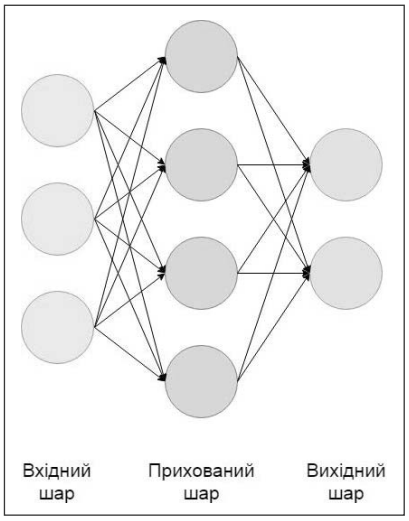

Рис. 1. Структура нейронної мережі

Етап розпізнавання тексту можна поділити на декілька етапів:

1. Обробка зображення та розпізнавання тексту у цілому. Цей етап нам необхідний для виявлення меж тексту. Ця процедура відбувається за допомогою роботи із фільтрами зображення для виявлення того, де знаходиться текст, а де зображені звичайні «шумові» предмети (ручка, олівець, цяточки на зображенні та ін.).

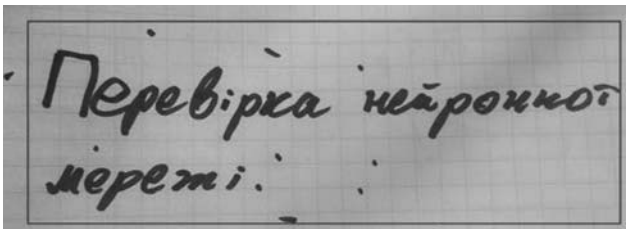

Рис. 2. Розпізнавання області тексту на зображенні

2. Обробка кожного окремого слова за допомогою визначення відступів між словами. Головною проблемою цього етапу може бути переплутування літери у слові з окремим словом. Для вирішення цієї проблеми нейронна мережа має навчитися пристосовуватися до почерку окремої людини.

Рис. 3. Розпізнавання кожного окремого слова у тексті

3. Виявлення окремих літер у слові та розпізнавання, на які печатні літери вони схожі. Складність цього етапу буде полягати тільки у почерку кожної людини та виявлення основних рис кожної окремої літери у слові.

Рис. 4. Розпізнавання кожної окремої літери у слові

Перевірка розпізнавання кожного окремого слова робиться за допомогою виявлення зміни кольору та встановлення коефіцієнта жирності тексту, щоб можна було виявити особливі прикмети кожної літери [3; 4; 5]. Наприклад, для літери «в» основними прикметами є два кола. Щоб знайти ці два кола, по-перше, потрібно виявити грані кожної літери. Наприклад, на зображенні можна побачити, де присутній напис ручкою, а де ні. По-друге, потрібно побудувати символ за допомогою лінії.

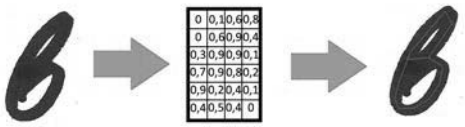

Рис. 5. Розпізнавання літери та її основних прикмет

Робота виявлення та розпізнавання кожної літери не така і складна, але бувають моменти, коли доводиться продумати, як можна розрізнити для нейронної мережі такі літери, як «Й» та «И» та інші. На жаль, такі моменти будуть виникати під час роботи системи доволі часто, на допомогу якраз і приходить етап виявлення кожного окремого слова у реченні, що дає можливість виділити цілу область слова та побачити, чи не пропустила нейронна мережа випадково який-небудь напис.

Загалом для коректної роботи потрібно зробити тренування для більш досконалішого алгоритму розпізнавання писемних символів. У кожної людини свій окремий почерк, в якому буває важко розпізнати, які літери написані, або текст може бути написаний настільки стисло, що нейронна мережа може розпізнавати окремі слова як одне слово [8; 9].

Для тренування нейронної мережі краще за все для людини написати декілька раз літери

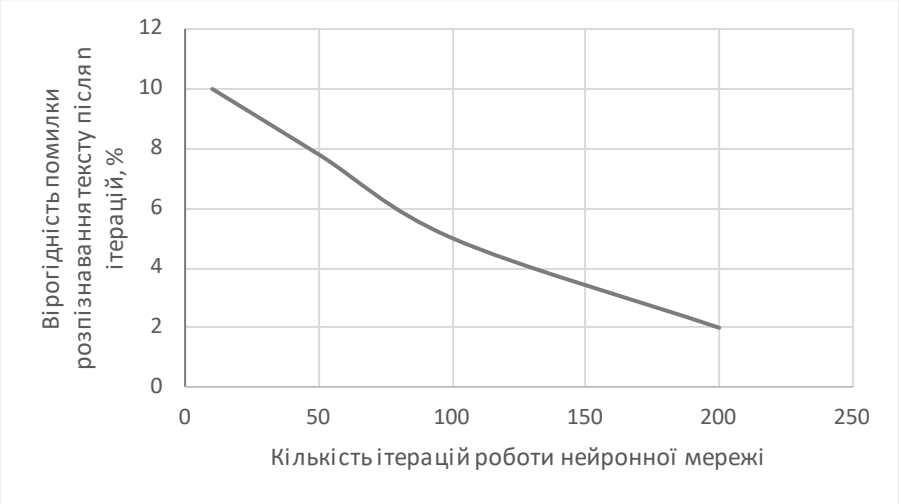

Рис. 6. Графік залежності процента помилок роботи нейронної мережі від кількості навчальних ітерацій

алфавіту, щоб нейронна мережа могла пристосуватися до почерку людини [6; 7]. Загалом з графіка ми можемо бачити, що зі збільшенням кількості використання нейронної мережі кількість помилок у розпізнаванні літер зменшується [10].

Особливим етапом має бути обробка тексту з помилками, де нейронна мережа мусить визначити ці помилки, виправити їх і правильно обробляти. Для цього до нейронної мережі потрібно підключити словник та під'єднати до цього правила граматики кожної мови. Наприклад, компанія Google протягом 10 років намагається зробити коректну роботу свого перекладача, але результат досі не найкращий, система іноді не може провідміняти деякі слова.

Для роботи нейронної мережі з помилковими вхідними даними кількість ітерацій роботи нейронної мережі має сягати 1000 та більше разів. Чим більша кількість ітерацій, тим краще нейронна мережа буде справлятися зі своєю роботою.

Висновки. Якщо говорити про експеримент, який був представлений у статті, то ми можемо бачити, що алгоритм розпізнавання тексту можна вдосконалювати за допомогою збільшення кількості випробувань такої системи. Важливу частину роботи такої системи відіграватиме оптимізація алгоритму обробки фотографії з текстом.

Робота алгоритму розпізнавання тексту може мати багато сфер, де вона має шанс бути використаною. Одразу можна сказати, що це роботи з розпізнавання тексту незнайомої для людини мови та її перекладом на знайому мову.

Загалом, якщо говорити про нейронні мережі, то вони мають перспективи використання в усіх сферах життя людини. Наприклад, якщо ми будемо говорити про проект розумного дому, де система буде підлаштовуватися під життя людини та обробляти такі вхідні дані, як час, температура та вологість повітря, голос людини та інше, а на виході результатом буде коректна робота такої системи, що може поліпшити життя людини.

Список літератури:

1. Кононюк А.Ю. Нейронні мережі і генетичні алгоритми. Київ, 2008. 446 с.
2. Хайкин С. Нейронные сети: полный курс. 2006. 1104 с.
3. Wang T., Wu D., Coates A. End-to-End Text Recognition with Convolutional Neural Networks. 2012. 60 p.
4. Bishop C. Pattern Recognition and Machine Learning. 2006. 758 p.
5. Raschka S. Python Machine Learning. 2005. 456 p.
6. Locascio N. Fundamentals of Deep Learning. 2017. 298 p.
7. Whitely D. Genetic Algorithms and Neural Networks: Optimizing Connections and Connectivity. 1990. 361 p.
8. Rashid T. Make Your Own Neural Network. 2016. 222 p.
9. Rogers J. Object-Oriented Neural Network in C++. 1997. 310 p.
10. Хинтон Д. Как обучаются нейронные сети. 1992. 107 p.

СИСТЕМА РАСПОЗНАВАНИЯ ПИСЬМЕННЫХ СИМВОЛОВ С ПОМОЩЬЮ НЕЙРОННОЙ СИСТЕМЫ

Нейронные системы находят все большее количество сфер, где они могут применяться, начиная с помощи в защите информации на мобильных устройствах и заканчивая алгоритмами прогнозирования каких-либо социальных, экономических или природных процессов. В статье будет раскрыт вопрос роботы одной из таких нейронных систем для определения рукописных символов. Во время анализа данного вопроса будет определено, как можно научить систему распознавать символы и как можно улучшить работу данного алгоритма с помощью самообучения нейронной системы. Также будет рассмотрен вопрос работы нейронной системы при некорректных входящих данных и какие перспективы работы такой системы в целом. В целом статья должна раскрыть вопрос предназначения нейронной системы и целесообразность её использования.

Ключевые слова: *нейронная система, нейрон, машинное обучение, алгоритм, система.*

RECOGNITION SYSTEM OF WRITING SYMBOLS BY NEURAL NETWORK

Neural systems find an increasing number of spheres where they can be applied. It can be use from helping to protect information on mobile devices and to create any prediction of any social, economic or natural processes. The article will try to answer on question of how neural system can determine handwritten symbols. During the analysis of this issue it will be determined how it is possible to teach the system to recognize symbols and how to improve the operation of this algorithm by self-learning algorithm of neural system. Also we will try answer on question of work neural network with incorrect input data and perspective of neural network. In whole the article need to answer the question of neural network mission and do we need to use it or we have another alternatives.

Key words: *neural system, neuron, machine learning, algorithm, system.*

УДК 681.51

Стародуб А.О.
Одеський національний політехнічний університет

Бабіч В.Ф.
Одеський національний політехнічний університет

РОЗРАХУНОК СТАТИЧНИХ ХАРАКТЕРИСТИК ПРОЦЕСУ ГОРІННЯ ГАЗОПОДІБНОГО ПАЛИВА ЯК ОБ'ЄКТА УПРАВЛІННЯ

У сучасних умовах витрати на паливо становлять значну частину бюджету теплопостачальних підприємств, особливо в зонах з помірним і холодним кліматом. Тому в умовах зростання цін на енергоносії та загострення екологічних проблем все більш високі вимоги висуваються до систем оптимізації використання енергії органічного палива. У статті представлена математична модель статики процесу горіння газоподібного палива як об'єкта управління. Представлені розрахункові формули та результати розрахунків у середовищі Matlab. Також отримані розрахунки концентрацій складників димових газів установки для згоряння газоподібного палива в режимах недостачі і надлишку повітря.

Ключові слова: *процес згоряння палива, об'єкт управління, оптимізація, модель динаміки, статичні характеристики.*

Постановка проблеми. Найбільшими забруднювачами навколишнього середовища шкідливими викидами техногенного характеру в атмосферу є продукти спалювання органічного палива в котлах теплових електростанцій, підприємств промислового виробництва, водогрійних і опалювальних установках житлово-комунального господарства, а також двигунами внутрішнього згоряння автотранспорту.

У більшості установок спалювання палива, що використовуються нині, оптимізація режиму горіння забезпечується шляхом підтримки співвідношення тиску палива і повітря відповідно до режимної карти. Такий спосіб є не досить ефективним, оскільки він не дає змогу враховувати зміни температури і вологості повітря, теплотворної здатності і температури газу і низки інших зовнішніх факторів. У зв'язку з цим у разі складання режимних карт допускають наявність значного надлишку повітря, щоб за жодних умов не допустити виникнення хімічного недопалу. У результаті в деяких режимах витрати повітря на горіння палива перевищують оптимальні в 1,5–2 рази, що збільшує витрату електроенергії на дуття і призводить до необхідності нагрівання надлишкового повітря, що подається в топкову камеру, тобто до додаткової витрати палива.

Аналіз останніх досліджень і публікацій. У спеціалізованій статті [1, с. 174–177] йдеться про перспективу використання мініатюрних датчиків для регулювання в технології спалювання палива. Також, аналізуючи дослідження із використання сучасних систем автоматичної оптимізації у більшості установок спалювання палива, що використовуються нині, оптимізація режиму горіння забезпечується шляхом підтримки співвідношення тиску палива і повітря відповідно до режимної карти, але такий спосіб є не досить ефективним.

Постановка завдання. Більшість розроблених систем автоматичної оптимізації співвідношення «паливо–повітря» побудовані із використанням стаціонарних газоаналізаторів і використовують коригуючий сигнал за величиною вмісту кисню в димових газах. На деяких типах котлів ці системи регулювання передбачені проектною документацією в обов'язковому порядку. Однак ці системи, як правило, не працюють у режимі регулювання, а газоаналізатор використовується в моніторинговому режимі, що зумовлено низкою причин:

– концентрація кисню в димових газах залежить не тільки від витрати повітря в топці (дуття), але й від інших умов експлуатації (неконтрольованого підсмоктування повітря, що спотворює показання киснеміра, зміни характеристик пальників, неідентичності характеристик пальників в багатопальникових котлах, зміни теплотворної здатності та виду палива, коливання вологості повітря тощо), що, своєю чергою, знижує ефективність роботи системи з регулювання за величиною вмісту кисню в димових газах;

– обмежене поширення мікропроцесорних контролерів, що мають стійкі та надійні алго-

ритми роботи з газоаналізаторами за значних збурень у широкому діапазоні навантаження установки, оскільки багато з розроблених алгоритмів регулювання не враховують перехідні процеси в топці у разі зміни потужності, а також інерційність електроприводів вентиляторів і димососів.

Вказані причини контрастують з вимогою, щоб у заданій продуктивності установки, тобто в кожній робочій точці і в будь-який момент часу оптимальна ефективність процесу згоряння палива була гарантована, мінімізуючи при цьому забруднюючі речовини, такі як сажа, CO, H_2, CnHm, а також CO_2. Виконання цих вимог стає дедалі важчим за дуже змінних граничних умов без відповідного інноваційного контролю горіння.

Тому необхідні нові рішення для керованої або оптимізованої експлуатації установок спалювання палива, які також можуть бути використані економічно в системах автоматичного регулювання установок від малої до великої потужності і навіть для регулювання двигунів внутрішнього згоряння. Прикладом прогресивних рішень у цьому напрямі є системи автоматичного регулювання процесу горіння в різноманітних установках для згоряння палива як постійного складу (сертифікованого), так і змінного складу (несертифікованого) з використанням коригуючих сигналів за вмістом компонентів хімічного недопалу H_2 і CO в димових газах [1, с. 175].

Виклад основного матеріалу дослідження. В сучасних умовах витрати на паливо становлять значну частину бюджету теплопостачальних підприємств, особливо в зонах з помірним і холодним кліматом. Тому в умовах зростання цін на енергоносії та загострення екологічних проблем все більш високі вимоги пред'являються до систем оптимізації використання енергії органічного палива.

Основними забруднювачами навколишнього середовища є продукти неповного згоряння, такі як сажа, CO, H2 та незгорілі вуглеводні CnHm. Зменшення викидів забруднюючих речовин можливе перш за все за рахунок розробки систем керування процесом горіння під час експлуатації установок згоряння палива. З цінових та технологічних причин вони майже не використовується в установках згоряння малої (до 200 кВт) та середньої потужності (до 3 МВт).

У ідеальному стехіометричному горінні ($\alpha = 1$) в установку подають з повітрям стільки кисню, скільки потрібно для повного спалювання палива. У димовому газі відсутній залишковий кисень, і котел працює з максимальною ефективністю. Нестача повітряного кисню призводить до втрат ефективності через неповне згоряння палива (появі в димових газах H_2, CnHm, NO) та високих викидів шкідливих речовин, таких як сажа, CO одночасно. Надлишок повітря в топці призводить до зайвих теплових витрат, оскільки не використане для горіння повітря переноситься як баласт і викидається в навколишнє середовище нагрітим до високої температури.

На рисунку 1 показаний типовий профіль статичних характеристик процесу згоряння палива – залежностей концентрацій основних компонентів димових газів на виході з котла від коефіцієнта надлишку повітря в топці котла $\alpha = 21 / (21 - O_2)$.

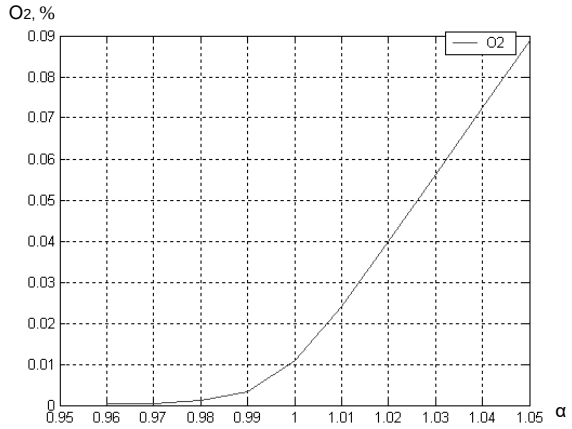

Рис. 1. Типові статичні характеристики процесу згоряння палива в котлі

Таким чином, як видно з рисунку 1, за підтримання оптимального коефіцієнта надлишку повітря в топці коефіцієнт корисної дії установки і температура в топці мають екстремуми. Ці залежності є типовими, що дає змогу складну і неоднозначну задачу екстремального регулювання замінити задачею підтримання на рівні слідів (сотих або тисячних об. процентів) концентрацій продуктів хімічного недопалу H_2 або CO в димових газах. Складність реалізації задачі автоматичної оптимізації процесу спалювання міститься у використанні високочутливих і стабільних датчиків концентрацій водню H_2 та окису вуглецю у високотемпературних потоках димових газів з високою вологістю.

Детальніше розглянемо алгоритм розрахунку статичних характеристик процесу горіння.

Кількісний склад продуктів згоряння визначається температурою, загальним тиском, під яким перебуває газова суміш, а також ваговими частками хімічних елементів, що входять у сполуки, що становлять продукти згоряння [3, с. 411].

Ступінь дисоціації газу швидко зростає зі збільшенням температури й залежить від тиску. Зі зниженням загального тиску в продуктах згоряння вуглеводного палива збільшується відносний вміст продуктів неповного згоряння й взагалі всіх продуктів, утворення яких супроводжується витратою тепла й збільшенням хімічної енергії, тобто ступінь дисоціації продуктів згоряння збільшується.

Розрахунок сполуки продуктів згоряння з урахуванням дисоціації починається зі складання таких рівнянь [3, с. 411]:

– рівнянь констант рівноваги тих реакцій, які враховуються в розрахунку;

– рівнянь балансу елементів, що входять у горючу суміш;

– рівняння повного тиску продуктів згоряння.

У горінні вуглеводнів у повітрі або кисні утворюються продукти згоряння, що містять тільки чотири елементи: вуглець, водень, кисень і азот. Тому стосовно цієї системи елементів приводиться методика розрахунку сполуки продуктів згоряння.

Рівняння реакцій дисоціації записуються у такому вигляді:

$$\left.\begin{aligned} & CO_2 \leftrightarrow CO + \frac{1}{2}\cdot O_2 \\ & K_1 = \frac{p_{CO}\, p_{O_2}^{0.5}}{p_{CO_2}} = f_1(T) \end{aligned}\right\} \quad (1)$$

$$\left.\begin{aligned} & H_2O \leftrightarrow H_2 + \frac{1}{2}O_2 \\ & K_2 = \frac{p_H\, p_{O_2}^{0.5}}{p_{H_2O}} = f_2(T) \end{aligned}\right\} \quad (2)$$

$$\left.\begin{aligned} & CO_2 + H_2 \leftrightarrow CO + H_2O \\ & K_3 = \frac{p_{CO}\, p_{H_2O}}{p_{CO_2}\cdot p_{H_2}} = f_3(T) \end{aligned}\right\} \quad (3)$$

$$\left.\begin{aligned} & N_2 + O_2 \leftrightarrow 2NO \\ & K_4 = \frac{p_{NO}^2}{p_{N_2}\, p_{O_2}} = f_4(T) \end{aligned}\right\} \quad (4)$$

$$\left.\begin{aligned} & H_2 \leftrightarrow 2H \\ & K_5 = \frac{p_H^2}{p_{H_2}} = f_5(T) \end{aligned}\right\} \quad (5)$$

$$\left.\begin{aligned} & O_2 \leftrightarrow 2O \\ & K_6 = \frac{p_O^2}{p_{O_2}} = f_6(T) \end{aligned}\right\} \quad (6)$$

$$\left.\begin{aligned} & N_2 \leftrightarrow 2N \\ & K_7 = \frac{p_N^2}{p_{N_2}} = f_7(T) \end{aligned}\right\}, \quad (7)$$

де K_i – константи рівноваги, що залежать тільки від температури й типу хімічної реакції.

Із використанням залежностей (1) – (7) та теплофізичних властивостей вуглецевоводневого палива і їх продуктів згоряння [4, с. 288] були розроблені алгоритм і програма для середовища Matlab 6.5, розраховані статичні характеристики процесу згоряння газоподібного палива для широкого діапазону зміни коефіцієнта надлишку повітря.

Отримавши результати розрахунку статичних характеристик, на рисунках 2–8 показана залежність складу продуктів згоряння, що характеризують економічність процесу згоряння, від коефіцієнта надлишку повітря, а також токсичність продуктів згоряння, відповідно, що розраховані для природного газу стандартного складу в об. процентах [4, с. 288]: CH_4 = 92,8 %; C_2H_6 = 3,9 %; C_3H_8 = 1%, C_4H_{10} = 0,4%, C_5H_{12} = 0,3 %, N_2 = 1,5 %, CO_2 = 0,1% і складу повітря: N_2 = 79%, O_2 = 21%.

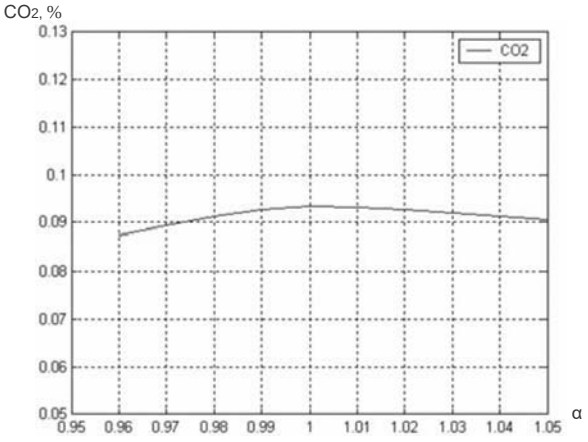

Рис. 2. Залежність вмісту кисню в продуктах згоряння від коефіцієнта надлишку повітря

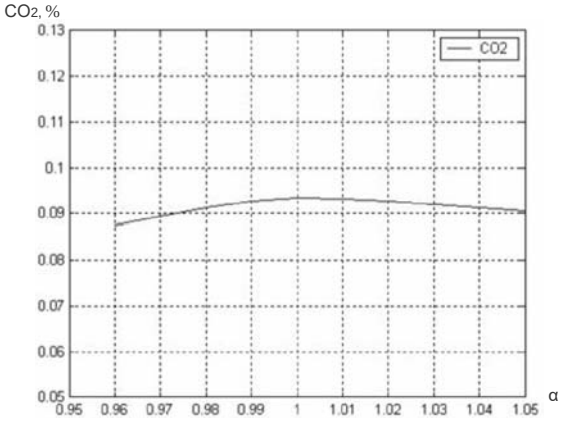

Рис. 3. Залежність вмісту діоксиду вуглецю в продуктах згоряння від коефіцієнта надлишку повітря

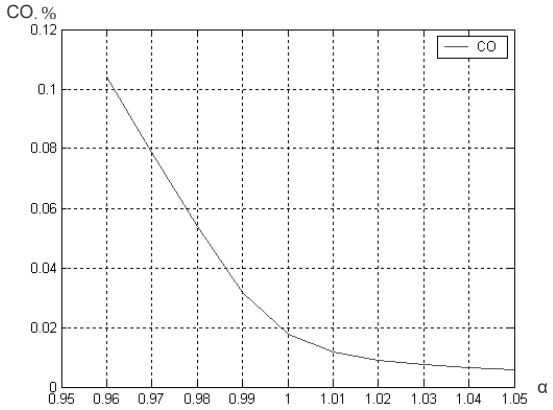

Рис. 4. Залежність вмісту оксиду вуглецю в продуктах згоряння від коефіцієнта надлишку повітря

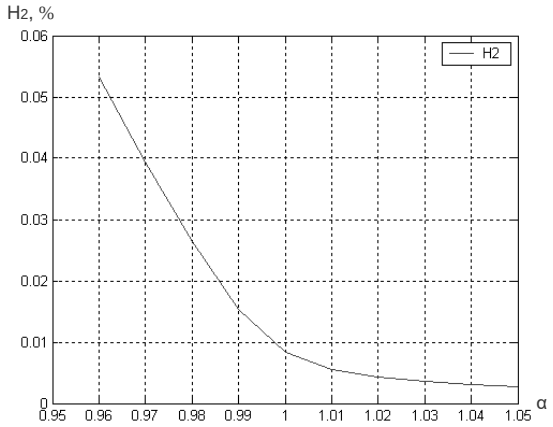

Рис. 5. Залежність вмісту водню в продуктах згоряння від коефіцієнта надлишку повітря

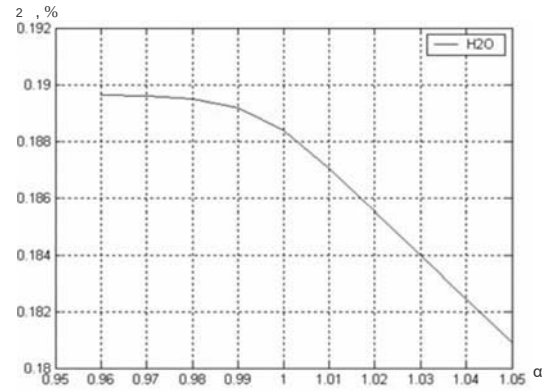

Рис. 6. Залежність вмісту водяних парів у продуктах згоряння від коефіцієнта надлишку повітря

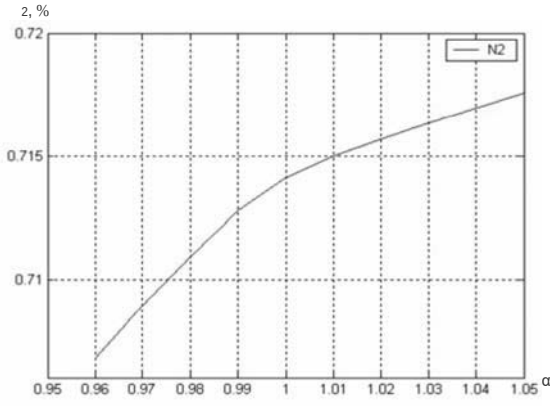

Рис. 7. Залежність вмісту азоту в продуктах згоряння від коефіцієнта надлишку повітря

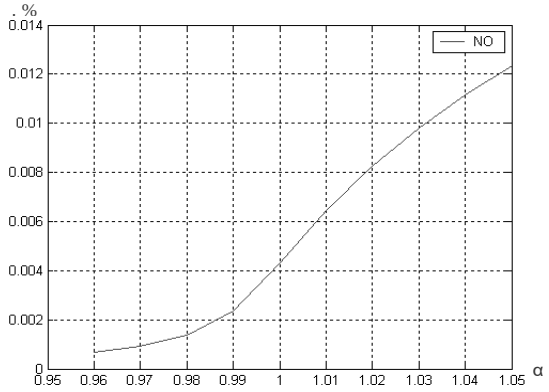

Рис. 8. Залежність вмісту NO в продуктах згоряння від коефіцієнта надлишку повітря

Висновки. У цій роботі були виконані розрахунки складу димових газів та побудовані кількісні залежності цих величин від коефіцієнта надлишку повітря (рисунки 2–8).

З рисунків видно, що вміст оксиду вуглецю CO і водню H_2 зменшується практично до нуля зі збільшенням коефіцієнта надлишку повітря, а вміст NO та кисню O_2 зростає. Вміст же діоксиду вуглецю CO_2 практично не залежить від зміни надлишку повітря.

Результати розрахунків дають змогу визначити коефіцієнти передачі моделі динаміки процесу спалювання палива у разі моделювання замкнутої системи регулювання співвідношення «паливо–повітря» з різними коригуючими сигналами як «у малому», так і «у великому», що дає змогу вибрати найбільш інформативний коригуючий сигнал.

Список літератури:

1. Völkel M., Hammer F. COe-Regelung mit Miniatur-Sensor eröffnet neue Perspektiven in der Verbrennungstechnik, Fachbeitrag, Gaswärme International (54). 2005. Nr. 3. S. 174-177.

2. Бабич В.Ф. Управление процессом термического обезвреживания промышленных сточных вод в вихревом аппарате для безотходного производства: автореф. дисс. канд. техн. наук. Одесса, 1986. 25 с.

3. Равич М.Б. Упрощенная методика теплотехнических расчетов. Москва: «Наука», 1966. 411 с.

4. Дубовкин Н.Ф. Справочник по теплофизическим свойствам углеводородных топлив и их продуктов сгорания. Москва, Ленинград: «Госэнергоиздат», 1962. 288 с.

РАСЧЕТ СТАТИЧЕСКИХ ХАРАКТЕРИСТИК ПРОЦЕССА ГОРЕНИЯ ГАЗОПОДОБНОГО ТОПЛИВА КАК ОБЪЕКТА УПРАВЛЕНИЯ

В современных условиях затраты на топливо составляют значительную часть бюджета теплоснабжающих предприятий, особенно в зонах с умеренным и холодным климатом. Поэтому в условиях роста цен на энергоносители и обострения экологических проблем все более высокие требования предъявляются к системам оптимизации использования энергии органического топлива. В статье представлена математическая модель статики процесса горения газообразного топлива как объекта управления. Представлены расчетные формулы и результаты расчетов в среде Matlab. Также получены расчёты концентраций составляющих дымовых газов установки для сгорания газообразного топлива в режимах нехватки и избытка воздуха.

Ключевые слова: *процесс сгорания топлива, объект управления, оптимизация, модель динамики, статичные характеристики.*

CALCULATION OF STATIC CHARACTERISTICS OF THE COMBUSTION OF GAS FUELS AS A CONTROL OBJECT

In modern conditions fuel costs constitute a significant part of the budget of heat supply enterprises, especially in temperate and cold climates. Therefore, in the conditions of rising energy prices and exacerbation of environmental problems, ever increasing demands are made on systems for optimizing the use of energy from organic fuels. The mathematical model of the statics of the combustion process of gaseous fuel as a control object is presented in the article. Presented calculation formulas and results of calculations in the Matlab environment. Calculations have also been made of the concentrations of the constituent flue gases for the combustion of gaseous fuels under conditions of shortage and excess air.

Key words: *process of combustion of fuel, object of control, optimization, model of dynamics, static characteristics.*

УДК 004.4

Сугоняк І.І.
Житомирський державний технологічний університет

Марчук Г.В.
Житомирський державний технологічний університет

Бобровнік С.О.
Human Interface Technology

СИНТАКСИЧНИЙ АНАЛІЗ КОДУ ДЛЯ СИСТЕМИ ДИСТАНЦІЙНОГО НАВЧАННЯ ПРОГРАМУВАННЯ НА МОВІ C#

Дослідження присвячене розробці алгоритму аналізатора коду для системи дистанційного навчання мов програмування. Отриманий результат полягає у модифікації методу синтаксичного аналізу коду на основі ітеративного алгоритму. Також у роботі побудовано модель синтаксичного аналізатора з розподілом лексем. Ця модель є математичним забезпеченням програмного комплексу дистанційного навчання програмування. Окрім того, в роботі викладено підхід до реалізації візуальних розподілених навчальних комплексів із використанням COTS-архітектури.

Ключові слова: *синтаксичний аналізатор, ітеративний алгоритм, аналіз програмного коду, дистанційне навчання, C#.*

Актуальність теми. З часу впровадження перших програмованих машин було створено понад дві з половиною тисячі мов програмування і з кожним роком кількість їх збільшується. Деякими мовами вміє користуватись тільки невелике число їх власних розробників, інші стають відомі мільйонам людей. Актуальність теми дослідження зумовлена тим, що для успішного та доступного навчання мов програмування потрібно реалізувати зручний візуальний навчальний комплекс з подальшою оцінкою успішності.

Метою дослідження є аналіз та побудова алгоритмів та моделей аналізатора коду для розробки системи дистанційного навчання мов програмування з коректною автоматичною оцінкою прогресу користувача. Основними завданнями, вирішеними в дослідженні, є: проведення аналізу теоретичних засад проектування та реалізації візуального навчального комплексу; розробка математичної та алгоритмічної моделі функціонування візуального навчального комплексу; реалізація програмної системи візуального навчального комплексу «Програмування C#».

Аналіз останніх досліджень і публікацій. Дослідженню роботи компіляторів та алгоритмів синтаксичного аналізу присвячено багато досліджень у зарубіжній науці такими науковцями, як: Альфред Ахо [1], Джеффрі Ульман, Раві Сеті, Ніклаус Вірт [2], Юрг Гуткнехт, Дик Грун, Робин Хантер. Слід звернути увагу на статтю [3] українського науковця В.І. Салапатова, де розглядається синтаксичний аналіз із розподілом лексем на групи. Такий підхід дає змогу значно спростити синтаксичний аналіз операторів мови та прискорити його виконання.

Отримані результати полягають у модифікації методу синтаксичного аналізу коду з використанням класичних алгоритмів та побудові моделі синтаксичного аналізатора з розподілом лексем на групи, який працює швидше, ніж класичні алгоритми через направлений перебір варіантів. Практичне значення роботи полягає у реалізації методів оцінювання прогресу навчання та синтаксичного аналізу текстів коду; розробці візуального навчального комплексу «Програмування C#» з оптимальними методами оцінювання.

Основна частина

Синтаксичний аналіз (parsing) – це процес співставлення лінійної послідовності лексем мови з його формальною граматикою. Під час синтаксичного аналізу текст оформлюється у структуру даних, зазвичай в дерево, яке відповідає синтаксичній структурі вхідної послідовності і добре підходить для подальшої обробки. Найчастіше синтаксичні аналізатори працюють у два етапи: на першому ідентифікуються осмислені токени, на другому створюється дерево розбору.

Синтаксичний аналіз мови програмування заснований на контекстно-вільній граматиці, за допомогою якої можна визначити більшу частину синтаксичної структури мови програмування.

Під час дослідження розглянуто *ітеративний алгоритм*, який багаторазово виконує базові методи аналізу з уточненням результатів на кожній ітерації. Особливістю цього алгоритму є те, що він завершується за скінченне число ітерацій, а отримане рішення є повним. Ємність пам'яті, якої потребує цей метод, лінійно залежить від довжини ланцюжка, що аналізується, но час може виражатися експонентою. Серед переваг ітеративного алгоритму відзначимо можливість управляти точністю і обчислювальною складністю аналізу за рахунок обмеження числа ітерацій. Показано, що рішення, отримане на будь-який ітерації алгоритму, є повним і може використовуватися для виявлення дефектів.

Також розглянуто алгоритм *Ерлі*. Цей метод синтаксичного аналізу дає змогу для довільної КВ-граматики розібрати вхідний ланцюжок за час $O(n^3)$, використовуючи при цьому ємність пам'яті $O(n^2)$, де n – довжина вхідного ланцюжка. Якщо граматика однозначна, то час дорівнює n^2, для більшості граматик мов програмування алгоритм можна модифікувати так, щоб час та ємність стали лінійними функціями від довжини вхідного ланцюжка.

Такі алгоритми дають змогу доволі просто і коректно проводити синтаксичний аналіз коду.

Основною метою впровадження дистанційної форми навчання є швидке й зручне поширення знань, забезпечення доступності освіти всім верствам населення. Значною мірою ця мета реалізується за допомогою програмних засобів, побудованих на сучасних інформаційно-комунікаційних технологіях, які одержали загальну назву системи дистанційного навчання (СДН). До найпопулярніших СДН можна віднести: Lotus Learning Space; Blackboard Learning System; REDCLASS; «Клас ХПІ». Слід зазначити, що у всіх системах відбувається ручна перевірка написаного коду, внаслідок чого аналіз результатів є надзвичайно нечітким і може бути упередженим, що унеможливлює якісний та об'єктивний моніторинг компетентностей студента.

Моніторинг допомагає відстежувати якість засвоєних знань і вмінь у навчальному процесі. Тому основною метою розробки та впровадження візуального навчального комплексу «Програмування C#» є підвищення ефективності навчання.

Результатом реалізації поставленого завдання є розгорнутий візуальний навчальний додаток, що містить у своєму складі серверну структуру збереження та обробки даних, кросплатформенний багатокористувацький клієнтський додаток для реалізації функціональних можливостей.

Центральним складником обробки даних у навчальному додатку є синтаксичний аналіз коду виконання навчальних вправ мовою програмування C#.

Синтаксичні аналізатори, які зазвичай використовуються в компіляторах, підрозділяють на два типи: спадні (top-down) і висхідні (bottom-up). Перші будують дерево розбору зверху до низу – від кореня до листя, другі від листя до кореня дерева. В обох випадках розбір рядка відбувається зліва направо.

Існує безліч реалізацій синтаксичних аналізаторів для розбору різного роду граматик. У разі мов програмування для розбору контекстно-вільних граматик використовуються так звані LL(k) (Left Left) і LR(k) (Left Right) аналізатори.

Розглянемо на прикладі алгоритм розбору для LL(1)- граматики.

G_1: (1) S → aAS; (2) S → b; (3) A → a; (4) A → bSA

Керуюча таблиця алгоритму представлена на рисунку 1.

	a	b	e
S	aAS,1	B,2	помилка
A	A,3	bSA,4	помилка
a	викид	помилка	помилка
b	помилка	викид	помилка
$	помилка	помилка	допуск

Рис. 1. Керуюча таблиця алгоритму

За допомогою таблиці проаналізуємо такий ланцюжок abbab:

(abbab,S$,e) ├ (abbab, aAS$, 1) ├ (bbab, AS$, 1) ├ (bbab, bSAS$, 14) ├ (bab, SAS$, 14) ├ (bab, bAS$, 142) ├ (ab, AS$, 142) ├ (ab, aS$, 1423) ├ (b, S$, 1423) ├ (b, b$, 14232) ├ (e, $, 14232)

Очевидно, що 14232 – лівий розбір ланцюжка *abbab*.

Розглянуті алгоритми передбачають посимвольний перегляд тексту програми з кроком уперед. При цьому всі елементи оператора мають оброблятися за єдиним алгоритмом. Це вносить певні ускладнення, оскільки нетермінали, які мають бути присутніми у певних місцях правил граматики, описуються за своїми власними правилами. Ці правила мають вигляд:

G = f(T, N, P, Ns), де

– *Ns* – кореневий символ (спеціальний нетермінал);

– *T* – це множина термінальних символів (у мові програмування це константи, ідентифікатори, ключові слова, символи пунктуації);

– N – це множина нетермінальних символів (у нашому разі такі поняття, як вирази, оператори та окремі частки операторів).

У зв'язку з цим доречно розглядати граматику кожного нетермінала та його обробку окремо. Так, якщо згідно з граматикою у певному місці оператора має бути вираз, то обробку оператора мови в цій частині треба виконувати згідно з граматикою виразу, тобто окремим модулем. При цьому ознакою кінця виразу є або ключове слово з конструкції оператора, або спеціальні знаки-розподільники.

Поєднання в одному аналізі розбору оператора, розбору виразу та інших частин оператора ускладнює та уповільнює процес компіляції у цілому. Тому розробка нових ефективних засобів синтаксичного аналізу у мовах програмування на цей час є все ще актуальною.

Як наведено у [6], можна виділити дві великі групи лексем: лексеми-об'єкти (описують дані) та лексеми дії (дії над цими даними). Таким чином,

$$L \rightarrow Ld + La,$$

де L – множина всіх лексем, Ld – лексеми-об'єкти, La – лексеми дії.

Для вирішення поставленої проблеми використаємо модифікований алгоритм «знизу догори».

Таблиці ідентифікаторів використаємо для визначення лексем та ключових слів. Вирази описуються граматикою, яка співпадає з правилами арифметичних дій, тобто існує пріоритетність виконання операторів, урахування дужок, які змінюють пріоритетність дій, та деякі інші. Ознаками кінця виразу можуть бути як ключові слова операторів, так і спеціальні символи.

Лексичний аналізатор обробляє лексеми L_d під час їх появи, причому тип результату визначається L_a. Оскільки порядок їх обробки визначений пріоритетом, це значною мірою спрощує опис правил граматики.

В останню чергу виконується породження гілок дерева поточного оператора як результат обробки ключових слів (через найнижчий пріоритет). Стартовий символ N_s, тобто тип оператора мови, розпізнається через першу або другу лексему.

Лексеми дії мають стандартний за семантикою набір, що присутній в усіх мовах програмування, та спеціальний набір тільки для конкретної мови. Пріоритетність дій у виразах визначає порядок виконання цих дій. Якщо пріоритет поточної лексеми дії не перевищує пріоритет попередньої, то виконується обробка попередньої лексеми дії. Інакше обробка відкладається у стек. Зобразимо порядок обробки лексем за допомогою діаграми активності, зображеної на рисунку 2.

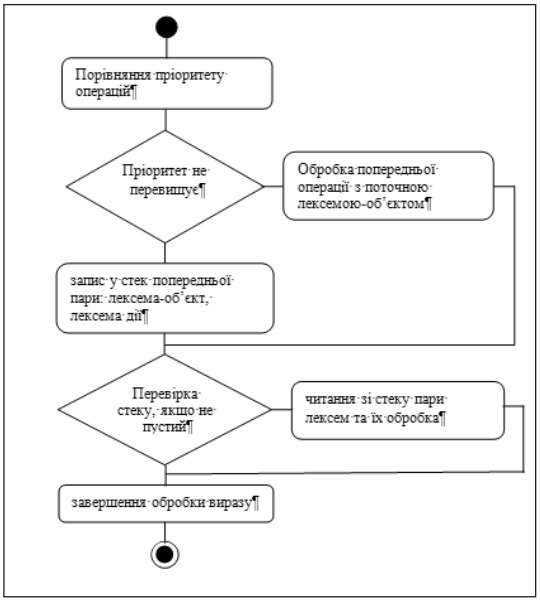

Рис. 2. Порядок обробки лексеми дій

Виконання алгоритму розподілу на лексеми реалізується механізмом доступу до стекової пам'яті. Якщо вираз має дужки, то виконання алгоритму починається із запису лівої дужки у стек з нульовим операндом і закінчується з появою правої. Права дужка ініціює зворотний хід у стеку, тобто обробку лексем у стеку з наступним їх виштовхуванням звідти. Обробка виразу закінчується, коли стек буде пустим. Результатом обробки є гілки синтаксичного дерева розбору.

Відсутність паритету дужок одразу ж виявляється через стек як помилка. У разі невідповідності типів операндів стандартний алгоритм має передбачати в разі необхідності перетворення типів (за допомогою спеціальних модулів). Тип результату виразу має відповідати опису оператора.

Розберемо вираз a + a * a для граматики G_2.

G_2: (1) E → E+T; (2) E → T; (3) T → T*F;
(4) T → F; (5) F → (E); (6) F →a.

Отже, відповідно до нашої граматики вираз a + a * a буде розпізнаний таким чином:

$(S, a + a * a, h_0) \vdash (S , a + a * a , h_0+) \vdash (S , a + a * a, h_0+) \vdash (S , a + a * a, h_0 +) \vdash (S, a + a * a, h_0 + a) \vdash (S, + a * a , h_0 +) \vdash (S, a * a, h_0) \vdash (S, a * a, h_0 *) \vdash (S, a * a, h_0 *) \vdash (S, a * a, h_0 * a) \vdash (S, * a, h_0 *) \vdash (S, a, h_0) \vdash (S, a, h_0a) \vdash (S, \$, h_0) \vdash (S, \$, \$).$

Послідовність правил, що була використана, відповідає лівому виводу вхідного ланцюжка:

$$E \Rightarrow E+T \Rightarrow T+T \Rightarrow F+T \Rightarrow a +T \Rightarrow a +T*F \Rightarrow a + F*F \Rightarrow a + a *F \Rightarrow a + a * a.$$

Дерево синтаксичного розбору буде мати вигляд, що представлений на рисунку 3. Побудова відбуватиметься зверху донизу.

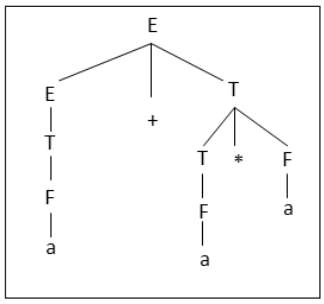

Рис. 3. Дерево синтаксичного розбору

За заданим алгоритмом опис граматики зводиться до опису ключових слів (зарезервованих і контекстних). Нині у версії 4.0 мови C # визначено 77 зарезервованих ключових слів і 18 контекстних. Оскільки лексеми дії можуть бути як спеціальними символами, так і ключовими словами, то код буде складатися з коду пріоритету та власне коду лексеми, а це певний тип обробки. Так, наприклад, ключове слово if породжує команди порівняння та відповідного умовного переходу. Всі ключові слова можна звести в одну таблицю та закодувати їх, щоб відрізнити лексеми дії виразу від лексем дії операторів.

Доречно робити розподіл обробки виразів та операторів. Окремим модулем робити обробку виразів, що на виході буде будувати дерево розбору виразу, а ключові слова операторів мови обробляти окремо.

Згідно з правилами граматики лексеми дії мають передбачати як стандартні, так і нестандартні дії. Всі дії задаються в таблиці ключових слів кодом, що є посиланням, що і забезпечує перехід на відповідну частину програми синтаксичного аналізатора.

Пошук термінала здійснюється у таблиці ключових слів, а тип нетермінала визначається типом лексеми, яка може бути оператором або виразом. Все це спрощує перевірку правильності синтаксису і дає змогу швидко виявити синтаксичні помилки.

Крім того, є стандартні оператори, такі як умовний оператор (if), оператори циклу (for, while, do… while), оператори введення/виведення, а є нестандартні (наприклад, побітові операції), до яких необхідно визначити правило обробки та додати модуль синтаксичного розбору. Звернення до них буде через посилання, яке представлено як код у таблиці ключових слів.

Ключові слова – це попередньо визначені зарезервовані ідентифікатори, які мають особливе синтаксичне значення, які мають спеціальні значення для компілятора і які мають певну послідовність в тій чи іншій конструкції оператора мови, обробка яких виконується наприкінці, тобто з найнижчим пріоритетом. Пріоритет використовується в основному при обробці виразів і досить зручно вписується у загальний алгоритм синтаксичного аналізу. У мовних конструкціях ключові слова операторів також можуть використовуватися як ознаки кінця виразу.

На етапі лексичного аналізу виконується розпізнавання не однієї лексеми, а пари лексем: лексеми-об'єкта та лексеми дії і передається далі синтаксичному аналізатору для подальшого розбору. На першому кроці аналізу оператора мови лексичний аналізатор розпізнає кореневий символ N_s, що визначає тип оператора, а подальший алгоритм розбору відтворюється згідно з правилами граматики.

Оскільки аналіз здійснюється через розбір пари лексем, то швидкість обробки операторів вища, ніж за класичними алгоритмами через направлений перебір варіантів, але ускладнюється алгоритм лексичного аналізу, а це впливає на загальну швидкість синтаксичного аналізу. Крім збільшення швидкості розбору, перевагою цього методу є спрощення правил опису граматики.

Крім того, в роботі розроблений алгоритм оцінювання прогресу студента, що базується на традиційних та нових (сучасних) методах контролю. До використаних традиційних методів контролю можна віднести тестування та практичні завдання. Якщо говорити про сучасні методи контролю, то використано кейс-вимірювачі, проекти, катанотести та контекстні завдання.

Завдання складається із декількох блоків, які допомагають студенту покращити свої знання в сфері програмування. Після проходження кожного блоку завдань студент проходить перевірку знань. Допоки студент не пройде попередній блок завдань наступний не буде доступний для навчання. Можна декілька разів проходити курс, з кожним разом покращуючи свої знання.

Блок завдань складається із декількох логічних рівнів:

- надання інформації по темі курсу;
- тестові завдання після закінчення курсу;
- практичне завдання після успішного проходження тестових завдань.

Тестові завдання додані з метою перевірки теоретичних знань студента, вони перевіряють знання основних положень курсу.

У вирішенні практичних завдань може бути декілька варіантів вирішення поставленої проблеми. Наприклад, якщо поставлено завдання від-

сортувати масив, студент може використати будь-який тип сортування, після написання алгоритму аналізаторами буде перевірено вихідний масив і виставлена оцінка.

Після успішного проходження блоку завдань студент отримує винагороду у вигляді 3D об'єкта, який можна поставити у віртуальному просторі за допомогою використання технології доповненої реальності.

Для переходу на новий рівень потрібно набрати вісімдесят чи більше відсотків успішних відповідей. Практичні і теоретичні завдання займають сімдесят та тридцять відсотків вартості відповідно.

У візуальному навчальному комплексі наявні такі теми:

– типи даних, літерали і змінні в мові програмування C#;
– оператори, масиви і рядки;
– введення в класи, об'єкти і методи;
– алгоритмічна підготовка студента з використанням псевдомови;
– наслідування, поліморфізм, інкапсуляція;
– інтерфейси, структури;
– делегати, події і LINQ;
– опрацювання виняткових ситуацій.

Під час навчання студент зможе покращити свої знання базових алгоритмів програмування, ознайомитися з Microsoft. Net Framework, C# та об'єктно орієнтованим програмуванням. Також під час програмування він розбереться з типами, зумовленими користувачем, структурами, класами, конструкторами, дізнається все про збирач сміття.

Для реалізації візуального навчального комплексу було використано COTS-архітектуру у взаємодії із об'єктно-центрованою моделлю (рис. 4). COTS – component off the shelf – методологія, яка дає розробникам можливість використання сторонніх компонентів.

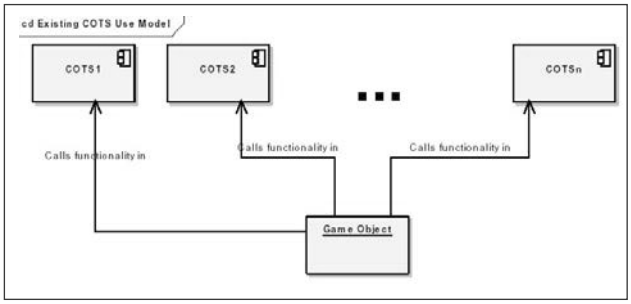

Рис. 4. COTS-архітектура у взаємодії із об'єктно-центрованою моделлю

Реалізація системи нараховує більше ста класів. У системі представлені класи, які описують операції з виведенням та проходженням завдань, графічною системою користувача, системою управління об'єктів та інше.

Для визначення спільного кордону функціональності системи та з метою створення основи

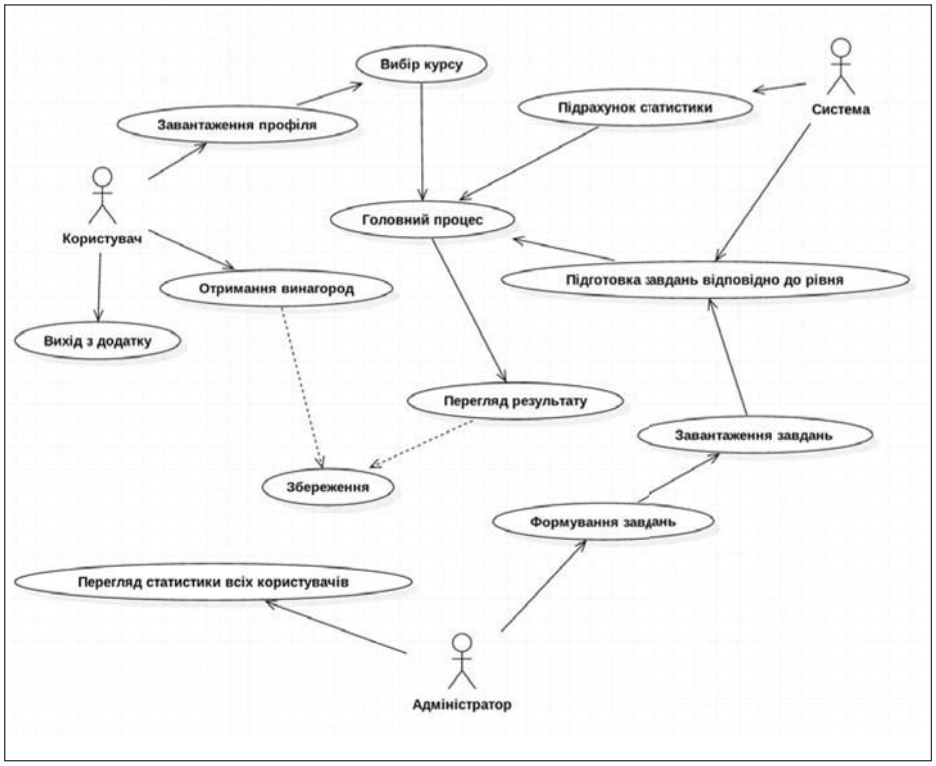

Рис. 5. Діаграма варіантів використання навчального комплексу