DOI: 10.36910/6775-2524-0560-2020-39-26
УДК: 004.624
Мельник Катерина Вікторівна, к.т.н., доцент,
https://orcid.org/0000-0002-9991-582X
Мельник Василь Михайлович, к.фіз.-мат.н., доцент,
https://orcid.org/0000-0001-8282-6639
Григоришин Андрій Миколайович, студент.
Луцький національний технічний університет, м. Луцьк, Україна.

АВТОМАТИЧНИЙ ЗБІР ІНФОРМАЦІЇ (ПАРСИНГ) В МЕРЕЖІ

Мельник К. В., Мельник В.М., Григоришин А. М. Автоматичний збір інформації (парсинг) в мережі. В даній роботі проаналізовано основні підходи до автоматизованого збору інформації в мережі інтернет, приведено характеристику контексту при зборі інформації. Розроблена система автоматичного збору інформації (парсингу) в мережі на мові програмування Golang.
Ключові слова: Парсинг, пошук інформації, Golang, автоматичний збір даних.

Мельник Е.В., Мельник В.М., Григоришин А. Н. Автоматический сбор информации (парсинг) в сети. В данной работе проанализированы основные подходы к автоматизированному сбору информации в сети интернет, приведена характеристика контекста при сборе информации. Разработана система автоматического сбора информации (парсинга) в сети на языке программирования Golang.
Ключевые слова: Парсинг, поиск информации, Golang, автоматический сбор данных.

Melnyk K., Melnyk V.M., Hryhoryshyn A. Automatic collection of information (parsing) in the network. This paper analyzes the main approaches to automated information collection on the Internet, and provides a description of the context for collecting information. A system for automatically collecting information (parsing) on the network in the Golang programming language.
Keywords: Parsing, information search, Golang, automatic data collection.

1. Постановка проблеми. Зміст поняття «пошук інформації» в мережі Інтернет в сучасній науковій літературі, в основному, зводиться до вивчення алгоритмів роботи пошукових систем. В сучасних умовах, всі дані, що представлені в глобальній мережі Інтернет, можна назвати неструктурованими. В основному, такі дані – це HTML сторінки, в яких зберігається текстова інформація, а також посилання на певні файли, що зберігаються на сервері. Через швидке зростання кількості інформації в мережі інтернет, виникла нагальна потреба розвивати технології, які дозволять збирати і обробляти інформацію швидко, цілеспрямовано і головне – автоматично. Розробка системи автоматичного збору інформації (парсингу) в мережі інтернет дозволить значно прискорити цей процес.[2, 3, 9]

Отже, парсер контенту – це не що інше, як скрипт, здатний сортувати інформацію, виділяючи найважливішу і обробляючи її згідно з алгоритмом, створеному для вирішення того чи іншого завдання. [1]

2. Аналіз останніх досліджень і публікацій. Пошук та аналіз текстової інформації досліджували видатні вітчизняні та зарубіжні вчені: Ньюелл А., Люгер Д. Ф., Фостер Д. М., Анісімов О. В., Поспєлов Д. О., Попов Є. В., Широков В. О.

Крім того, проблемі розробки системи автоматичного збору інформації (парсингу) в мережі приділяли увагу вітчизняні та зарубіжні вчені, зокрема: Цхошвілі Д. З., Іванова Н. А., Суханов, А. А., Маратканов, А. С., Менщиков А. А., Гатчин Ю. А., Каргін Б. Б., Логутова Т. Г., Єгорченкова Н. Ю., Єгорченков О. В., Чорна Н. О.

Парсинг сайтів є ефективним рішенням для автоматизації збору і зміни інформації. У порівнянні з людиною, комп'ютерна програма-парсер: швидко обійде тисячі веб-сторінок; акуратно відокремить технічну інформацію від «людської»; безпомилково відбере потрібне і відкине зайве; ефективно упакує кінцеві дані в необхідному вигляді.

3. Виділення невирішених раніше частин загальної проблеми. Виходячи з вищесказаного актуальним завданням є розробка методів персоналізації результатів роботи пошукових систем шляхом надання користувачу інструментів управління пошуковою видачею, а також використання нових моделей ранжування, заснованих на суб'єктивних для кожного користувача параметрах. Розробка програмного забезпечення (ПЗ), яке вирішує проблеми пошуку та автоматизованої обробки інформації, нині дуже потрібна й актуальна. Великі обсяги інформації змушують користувачів таких сайтів витрачати все більше і більше часу на її вивчення.[4, 6, 7]

4. Формулювання мети дослідження. **Мета дослідження** полягає проведенні аналізу існуючих методів автоматичного збору інформації та розробка власної системи автоматичного збору інформації (парсингу) в мережі.

5. Виклад основного матеріалу дослідження. Для збору інформації в мережі інтернет використовують спеціальні програми, які ще називаються веб-роботами або парсерами (краулерами). Їх можна умовно розділити на дві категорії: роботи, використовувані для законних цілей (аналіз контенту сайту, індексування для покращення роботи пошукових систем або створення «дзеркал» веб-сайтів) і програми, що використовуються зловмисниками для незаконних дій з порушенням авторських прав. Веб-роботи можуть не тільки збирати і обробляти інформацію, але і виконувати активні дії на веб-ресурсі, такі як купівля товарів і послуг, написання рекламних текстів, розсилка спаму. Крім того, робота веб-роботів призводить до збільшення навантаження на сервер і зменшення пропускної здатності, а також проблем доступу до ресурсу у звичайних користувачів.[9, 10]

В даний час в мережі інтернет доступні два типи парсерів: послуга парсингу ресурсів, що цікавляться такими програмами та використання універсальних програм-парсерів. У першому випадку замовник звертається із запитом на збір необхідних даних. В результаті він отримує шукану інформацію в необхідному вигляді і форматі [8]. Перевагами цього способу є: можливість отримати до 10000 записів; парсинг може проводитися будь-якого веб-ресурсу; немає необхідності проходити реєстрацію на сайті і використовувати VPN; надання інформації в зручному форматі. Недоліками даного способу є: висока вартість наданої послуги парсингу сайтів (ціна може варіюватися від 100$ за 10000 записів); необхідно тривалий час на збір даних (від 1 до 3 днів); додаткові витрати за необхідність доповнення або оновлення даних (від 50$).

Отже, парсер контенту – це не що інше, як скрипт, що здатний сортувати інформацію, виділяючи найважливішу і обробляючи її згідно з алгоритмом, створеному для вирішення того чи іншого завдання. Проаналізуємо, для чого можна використовувати парсер контенту.

По-перше, як відомо, найкращі сайти – це ті Інтернет-ресурси, на яких є цікава актуальна інформація. Нікому не потрібні вчорашні новини і сенсації, наприклад. Також, коли мова йде про сайти-обмінники валют, де необхідно змінювати інформацію про курс валют часом по кілька разів на день, створення парсера вкрай необхідно. Скрипт в таких випадках буде виконувати всю роботу сам, цілодобово відстежуючи зміни курсу валют в Нацбанку.

По-друге, парсер необхідний для автоматичного оновлення вашого Інтернет-ресурсу. Користувачі, які зайшли на сторінку один раз і виявили там застарілу інформацію, більше ніколи не повернуться на неї. Саме тому для збереження постійних користувачів, а також для залучення нових необхідне регулярне оновлення інформації на вашій інтернет-сторінці.

По-третє, парсер контенту – ідеальний інструмент для миттєвого наповнення сайту корисними даними. Так, коли в мережі величезна кількість сайтів різної спрямованості, лише мала їх частина виявляється в кінцевому підсумку корисними користувачам. Саме тому, важливо, щоб інформація на вашій веб-сторінці була не тільки актуальною, але ще і корисною.

По-четверте, – централізація даних. Відомо, що інформації в мережі предостатньо, при цьому самої різної. Вся проблема в тому, що вся ця інформація розкидана по безлічі Інтернет-ресурсах і зібрати її не так просто. Використовуючи парсер контенту, можна об'єднати вичерпну кількість корисної інформації на сайті, тим самим залучаючи все більше відвідувачів. Такий хід – відмінний варіант для далекоглядних розробників, які піклуються про те, щоб навіть випадкові відвідувачі, ставали постійними.

В процесі парсинга застосовуються скриптові мови програмування: PHP, Perl, Ruby, Python, JavaScript і багато інших. Лідерами ринку професійних інформаційно-пошукових систем є: система інтернет-моніторингу та конкурентної розвідки Avalanche, система пошуку інформації в Інтернеті FileForFiles & SiteSputnik і система автоматичного моніторингу веб-сайтів WebSite Watcher, система інтелектуального аналізу тексту «аналітичний кур'єр», систему аналізу текстової інформації RCO Fact Extractor SDK, систему аналізу текстових і структурованих даних PolyAnalyst.

Головні етапи пошуку інформації в мережі інтернет містять:

1. Аналіз процесу вилучення адрес. Алгоритм вилучення перебирає кілька варіантів і оцінює найкращий на основі найпростіших вагових характеристик і за допомогою експерта. Аналізуючи текст, програма має певний набір шаблонів, що виділяють певні блоки тексту, що містять якірні слова. Такий текст витягується із загального контексту з запасом в кілька слів спочатку і після кінця виразу.

2. Етап очищення тексту включає видаляються з тексту нерелевантних спеціальних символів, дужок, елементів вирівнювання або HTML розмітки. Також відбувається модифікація тексту з

урахуванням морфології, далі фраза розбивається на окремі слова і вирази та слова переводиться в початкову форму, яка придатна для подальшого пошуку в словнику.

3. Етап розбиття на слова. Текст після попереднього очищення розбивається на слова (токени) відповідно до регулярного виразу [!,?; / S] +. Додатково фільтруються зайві токени з чорного списку. Це допомагає максимізувати швидкість роботи системи за рахунок перебору тільки найбільш релевантних варіантів. Кожному токену зіставляється тип (числовий, текстовий, стрічковий).

В нашому дослідженні ми створимо та охарактеризуємо парсер на мові Golang.

Golang, або Go – мова програмування, початок якого був покладений в 2007 році співробітниками компанії Google: Робертом Гризмером, Робом Пайком і Кеном Томпсоном. Це швидка, статично типізована компільована мова, при використанні якої створюється враження використання динамично типизованої та інтерпретуємої мови. Розмір програм на Go розпочинається з 3 рядків і може досягати декількох мільйонів, записаних в один або декілька файлів з розширенням. go. Сучасні текстові редактори, наприклад, vim, підтримують його синтаксис. [5]

Go підтримує типобезпеку та можливість динамічного Введення даних, а також містить багату стандартну бібліотеку функцій і вбудовані типи даних на зразок масивів з динамічним розміром і асоціативних масивів. За допомогою механізмів багатопоточності Go спрощує розподіл обчислень і мережевої взаємодії, а сучасні типи даних відкривають програмісту світ гнучкого і модульного коду. Програма швидко компілюється, при цьому є збирач сміття і підтримується рефлексія.

Опишемо порядок налаштування оточення. Спершу викачаємо 64-бітову версію Go з офіційного сайту. Залежно від використовуваної операційної системи виконуємо наступні дії.

UNIX / Linux / MacOS X / FreeBSD

Витягаємо викачаний архів в теку / usr / local / go. Наприклад:

```
tar - C / usr / local - xzf go1.8.3. linux - amd64. tar. gz
```

Додаємо теку / usr / local / go / bin в змінну оточення PATH:

```
export PATH=$PATH: / usr / local / go / bin
```

Кожна програма на мові Go складається з пакетів (packages). Пакет main – головний, з нього розпочинається виконання програми.

Розглянемо головні етапи парсингу на Go:

1. Отримання початкового коду веб-сторінки – скачати програмний код тієї сторінки сайту, з якої необхідно витягти інформацію. У різних мовах для цього передбачені різні способи. В PHP найчастіше використовують бібліотеку cURL або ж вбудовану функцію file_get_contents. В Golang для цього підійде бібліотека net/http та функція get. При перегляді коду елемента на потрібній сторінці ми побачимо наступне (рис. 1).

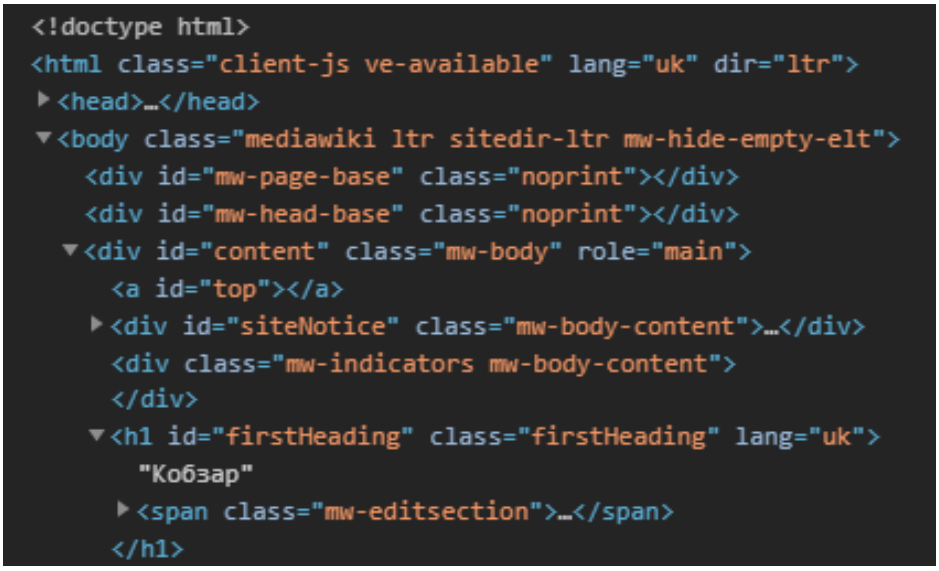

Рис. 1. Скріншот коду елемента Інтернет-сторінки

2. Витяг з html-коду необхідних даних. Отримавши сторінку, необхідно обробити її – виокремити із сторінки саме ту інформацію, заради якої і відбувається весь цей процес. Можна, звичайно ж, використовувати для цього регулярні вирази, проте є більш зручний шлях –

спеціалізовані бібліотеки. В Golang з цим справиться бібліотека gokogiri. Розглянемо дану бібліотеку на прикладі роботи розробленого парсера:

```
package main
import (
  "net/http"
  "io/ioutil
  "github.com/moovweb/gokogiri"
)
func main() {
  resp, _ := http.Get("http://www.google.com")
  page, _ := ioutil.ReadAll(resp.Body)
  doc, _ := gokogiri.ParseHtml(page)
  doc.Free()
}
```

Приклад реалізації парсера, можна знайти за посиланням: https://ahryhory-parser-go-example.online.

Скористатися цим сервісом досить легко. У верхній частині вікна є поле для введення тексту, в яке можна скопіювати посилання на сторінку з потрібною нам статтею (рис. 2).

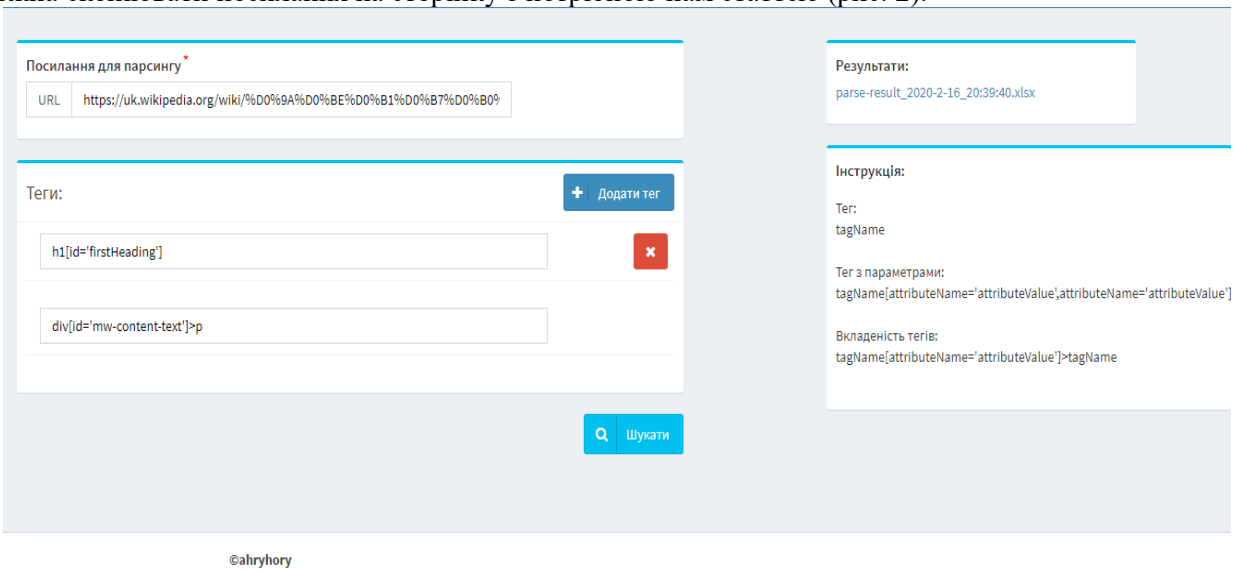

Рис. 2. Інтерфейс прикладу реалізації розробленого парсера

У блоці «Теги» (рис. 2) потрібно вписати вирази за якими буде здійснено пошук за заданим посиланням. Проведемо аналіз основних моментів цих виразів на основі прикладу, tagName[attributeName='attributeValue']>tagName:

1) «tagName» – назва потрібного тегу;
2) «attributeName» – назва атрибута в шуканому тегові;
3) «attributeName» – значення атрибута в шуканому тегові;
4) «>» – означає вкладеність, тобто наступний тег має бути всередині попередньо зазначеного;

3. Фіксація результату. Успішно оброблені дані на сторінці потрібно зберегти в необхідному вигляді для подальшої обробки. Спарсені дані зазвичай заносяться в базу даних, однак є й інші варіанти. Іноді потрібно записати в файл або будувати ієрархічні JSON-структури, іноді конвертувати в excel-таблицю або згенерувати динамічний rss-потік. Якщо пошукові параметри парсера (https://ahryhory-parser-go-example.online) були задані правильно, то у правій частині вікна (рис. 2) з'явиться доступний до скачування файл (.xlsx) з результатами (рис. 3).

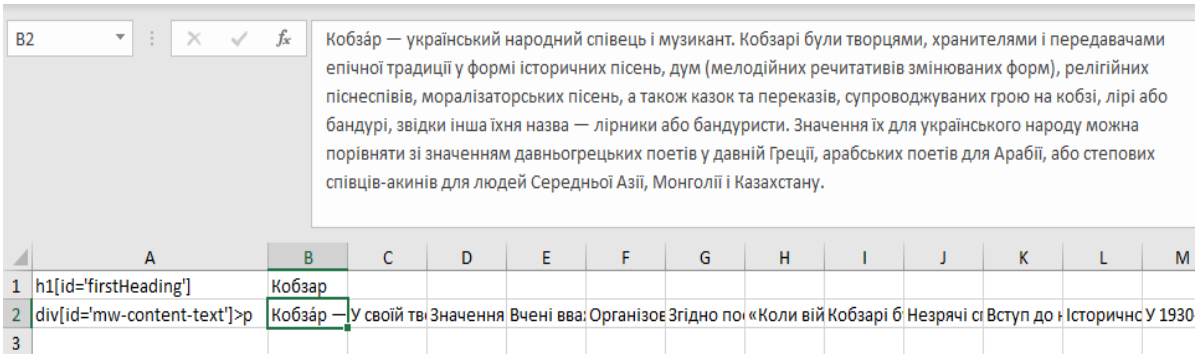

Рис. 3. Результати роботи розробленого парсера

6. Висновки. Нами проаналізовано основні підходи до автоматизованого збору інформації в мережі інтернет. Приведено характеристику контексту при зборі інформації і сказано, що отримання нової інформації безпосередньо з первинних джерел відбувається набагато рідше, тільки у випадках, коли пошук у відкритих джерелах виявляється марний або є непридатним від самого початку.

Перелічено основні етапи пошуку інформації в мережі інтернет та дано їх коротку характеристику. Окремо проаналізовано методи автоматизованого пошуку інформації. Для кожної програми наведено переваги та недоліки.

Наукова новизна дослідження полягає в розробці системи автоматичного збору інформації (парсингу) в мережі. Наголошено на основних інструментів та етапів для написання парсера та дана коротка характеристика мови програмування Golang. Перелічено існуючі проблеми при проведенні пошуку інформації сучасних умовах та перелічено основні методи пошуку.

Перспективи подальших досліджень. В перспективі подальших досліджень є розробка графічного інтерфейсу для сервера. Розроблена система надає користувачу ефективні, зручні та прості у використанні інструменти обробки веб-даних на всіх етапах взаємодії з семантичною павутиною від створення веб-ресурсів до пошуку інформації. Створені алгоритми парсинга можуть використовуватися програмістами у створенні власних програмних продуктів.

Список бібліографічного опису

1. Абрамова Т. А. Розробка парсинг-системи для отримання прихованих посилань зі сторінок соціальних мереж *Вісник ПензГУ. 2016. №3 (15)*. С. 22–31.
2. Вдовін І. В. Застосування технології Web Mining до вилучення наукових даних в мережі інтернет *Інформаційні технології в науці і виробництві: матеріали всерос. молодеж. науч. – техн. конф. Омськ: Вид-во ОмГТУ, 2015.* С. 10–14.
3. Вдовін І. В. Актуальні питання автоматичного вилучення даних з веб-сторінок *перспективи розвитку інформаційних технологій. 30 січня 2015 Новосибірськ, 2015.* С. 11–16.
4. Комарова А. В., Менщиков А. А. Метод автоматизованого вилучення адрес з неструктурованих текстів *International Journal of Open Information Technologies. 2017. №11*. С. 43–55.
5. Менщиков А. А., Гатчин Ю. А. Побудова системи виявлення автоматизованого збору інформації з веб-ресурсів *інженерні кадри – Майбутнє інноваційної економіки: в 8 ч. 2015. Т. 4.* С. 58–61.
6. Менщиков А. А. Вивчення поведінки засобів автоматизованого збору інформації з веб-ресурсів *Питання кібербезпеки. 2017*. №3 (21). С. 49–56.
7. Мосалев П. М. Огляд методів нечіткого пошуку текстової інформації *Вісник Московського державного університету друку. 2016*. № 2. С. 87–91.
8. Суханов А. А., Маратканов, А. С. аналіз способів збору соціальних даних з мережі Інтернет *International Scientific Review. 2017. Вип. 1.* С. 22–25.
9. Чиркин Е. С. Деякі проблеми автоматизованого вилучення даних з веб-сторінок *Інтернет і сучасне суспільство. Санкт-Петербург, 2015.* С. 291–294.
10. Madaan, A. Hadoop: Solution to Unstructured Data Handling *Big Data Analytics. Advances in Intelligent Systems and Computing. 2017. № 1.* P. 47–54.
11. Skobelev, V. V. Analysis of the structure of attributed transition systems without hidden transitions *Cybernetics and Systems Analysis. 2017. № 2.* P. 165–170.

References

1. Abramova T. A. Rozrobka parsynh-systemy dlja otrymannja pryxovanyx posylan′ zi storinok social′nyx merež *Visnyk PenzHU. 2016. #3 (15)*. S. 22–31.
2. Vdovin I. V. Zastosuvannja texnolohiji Web Mining do vylučennja naukovyx danyx v mereži internet *Informacijni texnolohiji v nauci i vyrobnyctvi: materialy vseros. molodež. nauč. – texn. konf. Oms'k: Vyd-vo OmHTU, 2015.* S. 10–14.

3. Vdovin I. V. Aktual′ni pytannja avtomatyčnoho vylučennja danyx z veb-storinok *perspektyvy rozvytku informacijnyx texnolohij. 30 sičnja 2015 Novosybirs'k, 2015.* S. 11–16.

4. Komarova A. V., Menščykov A. A. Metod avtomatyzovanoho vylučennja adres z nestrukturovanyx tekstiv *International Journal of Open Information Technologies. 2017. #11.* S. 43–55.

5. Menščykov A. A., Hatčyn Ju. A. Pobudova systemy vyjavlennja avtomatyzovanoho zboru informaciji z veb-resursiv *inženerni kadry – Majbutnje innovacijnoji ekonomiky: v 8 č. 2015. T. 4*. S. 58–61.

6. Menščykov A. A. Vyvčennja povedinky zasobiv avtomatyzovanoho zboru informaciji z veb-resursiv *Pytannja kiberbezpeky. 2017. #3 (21).* S. 49–56.

7. Mosalev P. M. Ohljad metodiv nečitkoho pošuku tekstovoji informaciji *Visnyk Moskovs'koho deržavnoho universytetu druku. 2016. # 2.* S. 87–91.

8. Suxanov A. A., Maratkanov, A. S. analiz sposobiv zboru social′nyx danyx z mereži Internet *International Scientific Review. 2017. Vyp. 1.* S. 22–25.

9. Čyrkyn E. S. Dejaki problemy avtomatyzovanoho vylučennja danyx z veb-storinok *Internet i sučasne suspil'stvo. Sankt-Peterburh, 2015.* S. 291–294.

10. Madaan, A. Hadoop: Solution to Unstructured Data Handling *Big Data Analytics. Advances in Intelligent Systems and Computing. 2017. № 1.* P. 47–54.

11. Skobelev, V. V. Analysis of the structure of attributed transition systems without hidden transitions *Cybernetics and Systems Analysis. 2017. № 2.* P. 165–170.