ОГЛАВЛЕНИЕ

УДК 004.051, 004.912

ОБЗОР СУЩЕСТВУЮЩИХ ИНСТРУМЕНТОВ ВЫЯВЛЕНИЯ ПЛАГИАТА И САМОПЛАГИАТА

А. Э. Тлитова[1], А. С.Тощев[2]

[1,2]*GDC Services*

[1]soloadelante15@gmail.com, [2]atoschev@kpfu.ru

Аннотация

Ученым необходимо всё время выпускать в свет результаты своих работ, чтобы оставаться востребованными, соответствовать времени, критериям и не оказаться вне научного общества. Известный принцип «публикуйся, или погибнешь» («Publish or Perish») чаще всего вынуждает ученых стремиться к количеству, а не к качеству [1]. Наряду с проблемами авторства, проплаченных исследований, фабрикацией результатов одними из распространенных нарушений являются плагиат и самоплагиат. Их воздействие является более тонким, но не менее разрушительным для научного общества.

В статье дан обзор существующих инструментов выявления заимствований в научных статьях авторов. Анализ решений выполнен путем сравнения систем по ряду характеристик. Для оценки работоспособности и эффективности созданных инструментов они протестированы на реальных данных.

***Ключевые слова**: плагиат, самоплагиат, научная этика, текстовые заимствования, анализ текстов*

ВВЕДЕНИЕ

Доверие к ученым степеням подрывается в связи с большим числом подделок и прецедентов самоплагиата. Отечественная наука могла бы развиваться эффективнее, так как KPI зависели бы не от количества опубликованных статей, а от их качества. Сфальсифицированные научные статьи также негативно влияют на экономическую сторону науки, так как другим ученым приходится тратить средства из государственного финансирования на то, чтобы выяснить, что данное исследование не проводилось, вопрос не освещался ранее или что на данную работу невозможно ссылаться в дальнейших исследованиях [2].

В то время, как плагиату посвящено много статей и исследований, проблема самоплагиата несправедливо обойдена стороной, хотя в наше время эта проблема всё чаще и чаще возникает в научных кругах. Число публикаций оказывает большое влияние на карьеру и достаток ученых, и, чтобы добиться успеха в виде повышения степени и признания, недобросовестные авторы повторно публикуют свои работы, не указывая, что эта информация уже была опубликована ранее. Повторяться могут как вся работа с незначительными изменениями, например, в названии, аннотации (двойная публикация), так и отрывки из предыдущих (нарезка салями). Под термином «самоплагиат» обычно подразумевается вторичное употребление автором своих разработок под видом новых [3]. Такие работы научными уже назвать нельзя. Наглядным примером может служить переизданный текст.

При самоплагиате автор нарушает свои обязательства перед тем журналом, в котором материал был опубликован в первый раз, так как при приеме статьи обычно происходит заключение договора, согласно которому на определенный срок право на исключительную публикацию научной работы передается конкретному журналу. При этом автор сохраняет право применять полностью или частично текст работы в других местах, но только при расположении ссылки на неё. Несоблюдение договора может привести к наказаниям в виде несогласия журнала размещать статью или отказа от сотрудничества с автором [4].

IEEE определяет плагиат как повторное использование чужих предыдущих результатов или слов без явного признания первоначального автора и источника [5]. Плагиат в любой форме недопустим и считается серьезным нарушением профессионального поведения с этическими и правовыми последствиями.

Согласно политике IEEE есть несколько базовых факторов, которые учитываются при оценке возможного плагиата [6]:

- количество (полная статья, раздел статьи, страница, абзац, предложение, фразы),
- использование кавычек для всего скопированного текста,
- надлежащее размещение ссылок на источники заимствования и т. д.

Дублирование информации без ссылок на источники недопустимо, даже если она является собственностью автора, так как это ставит под вопрос актуальность и научную новизну идеи.

Нередко для выявления самоплагиата используют системы обнаружения плагиата. Эти инструменты рассмотрены в данной работе.

Таким образом, целью данного исследования являются анализ существующих программ для выявления текстовых заимствований по ряду характеристик, изучение и тестирование действующих систем, выявление их проблемных областей.

МЕТОДОЛОГИЯ

В ходе исследования изучены существующие статьи, направленные на изучение систем выявления самоплагиата и плагиата, проанализирована их актуальность. Для этого была составлена таблица, включающая как уже описанные в работах [7], [8] системы с некоторыми правками, так и дополнительно найденные через поисковые системы. Правки в найденных описаниях были сделаны в связи с обнаружением неточностей и устаревшей информации в процессе анализа каждого инструмента. Также в таблицу внесены результаты тестирования каждой системы на предварительно подготовленных текстах:

1. В формате .txt. Русскоязычный текст скопирован из работы [3]. Некоторые слова заменены синонимами. Размер текста – 47 слов. Источник работы – сеть интернет.
2. В формате .pdf. Страница 3 без изменений изъята из англоязычной научной статьи [9]. Данная страница выбрана по причине наличия формул и рисунка для тестирования способности программ к их распознаванию. Размер текста – 280 слов. Источник работы – научная социальная сеть Research Gate.
3. В формате .doc. Текст из предыдущего документа, переделанный в иной формат с сохранением формул, но без изображения. Размер текста – 270 слов.

Для чистоты эксперимента и более широкого охвата инструментов для тестирования были выбраны фрагменты из статей, а не целые статьи, так как многие программы имеют ограничения по объёму текста для проверки.

РЕЗУЛЬТАТЫ И ОБСУЖДЕНИЯ

Системы, производящие анализ текста на наличие заимствований, делятся на два вида – программное обеспечение с возможностью установки на персональный компьютер и онлайн-сервисы.

Проанализированы следующие инструменты в виде программного обеспечения:

1. eTXT Антиплагиат (ПО) [10];
2. Advego Plagiatus [11];
3. Double Content Finder [12];
4. Viper [13];
5. Плагиата.НЕТ [14];
6. Plagiarism Checker X [15];

а также инструменты в виде онлайн-ресурсов:

1. Dupli Checker [16];
2. PaperRater [17];
3. Plagiarisma.net [18];
4. PlagiarismChecker [19];
5. Plagium [20];
6. PlagTracker [21];
7. PlagScan [22];
8. Plagiarism Detector [23];
9. Content Watch [24];
10. Grammarly [25];
11. Docoloc [26];
12. Text.ru [27];
13. Антиплагиат [28];
14. eTXT Антиплагиат (онлайн-сервис) [29];
15. Turnitin [30];
16. Ithenticate [31];
17. Quetext [32];
18. Copyleaks [33];
19. Scribbr [34];
20. Strikeplagiarism [35];
21. Small SEO Tools [36].

Сравнение проводилось по следующим параметрам: название, страна, формат поддерживаемых файлов, языков, доступное число проверок, максимальный размер документа, метод сравнения текстов, необходимость регистрации для онлайн-ресурсов и результат проверок.

В ходе работы было выявлено, что программы для персональных компьютеров намного хуже справляются с поиском заимствований. Их общий недостаток – необходимость неоднократно вводить капчу для поисковых систем. Если отключить эту проверку, то может снизиться точность определения.

Хуже всего справилась программа Плагиата.НЕТ. Её показатель заимствования для всех выбранных текстов – 0%. Лучше всего обнаружила плагиат Advego Plagiatus и выдала следующие показатели:

1. для .txt – 93%;
2. для .pdf – 67%;
3. для .doc – 74%.

Что касается онлайн-ресурсов, предоставляющих бесплатные проверки документов, то здесь наихудшие результаты показали Dupli Checker, PaperRater, Grammarly и Антиплагиат.

Девять систем, показавших наилучшие результаты, представлены в таблице 1. Более полная информация по всем системам отражена в таблице 2.

Таблица 1. Результаты проверок систем с высокими показателями

Инструмент	**% плагиата .txt файла**	**% плагиата .pdf файла**	**% плагиата .doc файла**
Copyleaks	59	92	98
Plagiarisma.NET	65	43	54
Content Watch	83,6	не поддерживается	55,9
Plagiarism Detector	100	не поддерживается	0
Small SEO Tools	100	0	0
Text.ru	88	не поддерживается	0
Etxt Антиплагиат	80	не поддерживается	8
Quetext	65	не поддерживается	0
PlagScan	57,8	0	0

Лучший процент обнаружения заимствований по всем документам у Copyleaks, Content Watch и Plagiarisma.NET.

Для анализа текста Copyleaks применяет передовые технологии искусственного интеллекта, поддерживает многие форматы загружаемых файлов и все языки, интерпретируемые форматом Unicode. На сайте необходима регистрация. Далее для проверки в бесплатном режиме сервис предлагает 10 страниц в месяц.

Plagiarisma.NET использует инструменты поисковых систем Google и Bing. В режиме поиска Bing результаты были точнее. Сервис также поддерживает многие форматы файлов и распознает текст на более чем 190 языках. Для зарегистрированных пользователей доступен также поиск заимствований в Google, но всего 3 сравнения в день.

Content Watch применяет специально разработанный авторский алгоритм. Здесь можно анализировать текст только путем вставки в окно ввода. Сервис лучше всего показал себя при анализе русскоязычного текста – 83,6% плагиата.

Чтобы исследовать вероятность ложных срабатываний на данных трех системах, было решено проверить русскоязычный текст со 100% уникальностью, содержащий 63 слова. Получены следующие показатели процента заимствования:

1. Copyleaks – 0%;
2. Plagiarisma.NET –12%;
3. Content Watch – 15,6%.

Результаты говорят о том, что Copyleaks лучше остальных справляется с анализом контекста слов, тогда как Plagiarisma.NET и Content Watch анализируют единицы предложения вне контекста и считают плагиатом устойчивые словосочетания.

Таблица 2. Результаты проверок всех проанализированных систем

Название	**Поддерживаемый формат файлов**	**Алгоритм сравнения текстов**	**Результат проверки, % плагиата**	**Необходимость регистрации**
Advego Plagiatus	.pdf, .doc, .txt	инструменты поисковых систем (Yandex, Yahoo!, Bing, Google); алгоритм шинглов; алгоритм лексических	txt-фразы 49%, слова 93%; pdf-фразы 18%, слова 67%; doc-фразы 17%, слова 74%	нет

		совпадений; алгоритм псевдонуализации		
Double Content Finder	.txt	инструменты поисковых систем (Yandex)	поддержка была прекращена	нет
eTXT Antiplagiat software	.doc, .txt	алгоритм шинглов; инструменты поисковых систем (Google, Yandex)	txt – 38%; doc – 8%	нет
Plagiata.NET	.doc, .docx, .rtf, .txt	инструменты поисковых систем (Yandex, Google)	txt – 0%; text from doc – 0%	нет
Plagiarism Checker X	.doc, .docx, .rtf, .pdf, .txt	не упомянуто	txt – 0% pdf – 10% (150 слов); doc – 13% (150 слов)	нет
Viper	.doc, .docx, .pdf, .html, .odt, .rtf, .text, .s, .cs, .app, .java, .ptt, .pttx.	инструменты поисковых систем	платно	нет
Antiplagiat	.pdf, .txt; input field	документ делится на небольшие «кусочки», а поисковый модуль осуществляет поиск по общедоступным источникам российского сегмента Интернета, содержащим одну или несколько следующих частей	txt – 0%; text from doc – 3%; pdf – 2,75%	да
Content Watch	input field	комплексный авторский алгоритм (без использования шинглов)	txt – 83,6%; doc – 55,9%	нет
Copyleaks	.html, .txt, .pdf, .docx, .doc, .rtf, .xml, .pptx, .ppt,	продвинутая технология ИИ	txt – 59%; pdf – 92%; doc – 98%	да

	.odt, .chm, .epub, .odp, .ppsx, .pages, .xlsx, .xls, .csv, LaTeX, Image Types (.gif, .png, .bmp, .jpg, .jpeg)			
Docoloc	.doc, .docx, .pdf, .odt, .html, .rtf, .txt	не упомянуто	платно	да
Dupli Checker	docx, .txt; поле ввода	инструменты поисковых систем (Google, Yahoo!, MSN)	txt – 0%; doc – 0%	нет
eTXT Antiplagiat online	поле ввода	алгоритм шинглов	txt – 80%; text from doc – 8%	нет
Grammarly	.doc, .docx, .odt, .txt, .rtf; input field	не упомянуто	txt – 0%; doc – 0%	нет
Ithenticate	.doc, docx., Word XML, WordPerfect, PostScript, .ppt, .pptx, .pdf, .html, .rtf, .hpw, .odt, .txt	использует запатентованный алгоритм, который преобразует каждую представленную рукопись в «цифровой отпечаток», который сравнивается – как отпечаток пальца человека – с обширной базой данных, где можно обнаружить тонкие оттенки сходства	платно	да
PaperRater	только поле ввода бесплатно; .doc, .docx, .txt, .odt, .rtf	инструменты поисковых систем (Yahoo!, Bing, Google)	текст из doc – 0%	нет
Plagiarisma.NET	.html, .txt, .rtf, .doc, .docx, .pptx, .xlsx, .xls, .pdf, .odt, .epub, .fb2	инструменты поисковых систем (Google, Bing)	txt – 65%; pdf – 43% (Bing), 0% (Google); doc – 54%	да (чтобы включить поиск Google)

			(Bing); 0% (Google)	
PlagiarismChecker	только поле ввода	инструменты поисковых систем (Yahoo!, Google)	отсылает к результатам поиска Google	нет
Plagium	.doc, .docx, .pdf, .txt; input field	авторская технология: ввод текста делится на более мелкие «кусочки», которые сравниваются с веб-контентом (работа основана на патентованном движке Septet TX Miner, который использует передовые технологии поиска для глубокого анализа документов в общедоступной всемирной паутине или в частных репозиториях)	платно	да
PlagTracker	.doc, .txt; поле ввода	авторская технология	отсылает на пустую страницу	нет
PlagScan	все текстовые форматы; поле ввода	авторская технология индексирования, основанная на Apache Solr™ + инструменты поисковых систем (Yahoo!)	txt – 57,8%; pdf – 0%; doc – 0%	да
Plagiarism Detector	все текстовые форматы; поле ввода	инструменты поисковых систем; алгоритм, который игнорирует статистически	txt – 100%; pdf – ошибка "invalid hash key"; doc – 0%	нет

		общие фразы		
Quetext	только поле ввода бесплатно; .txt	алгорим, вычисляющий сходство на основе множества различных технических факторов и передовых методов машинного обучения (алгоритм DeepSearch ™ выходит за рамки простого сопоставления слов с тем, что называют «контекстным» плагиатом)	txt – 65%; doc – 0%	да
Scribbr	.doc, .docx, .pdf, .txt	не упомянуто	платно	да
Small SEO Tools	.doc, .docx., .pdf, .txt, URL	инструменты поисковых систем	txt – 100%; text from doc – 0%; pdf – 0%	нет
Strikeplagiarism	.doc, .docx, .rtf, .odt, .pdf	не упомянуто	платно	да
Text.ru	.doc, .docx, .odt, .rtf, .pdf, .htm, .html, .txt; input field	авторский алгоритм (без использования шинглов); инструменты поисковых систем	txt – 88%; text from doc – 0%	нет
Turnitin	.doc, .docx, .odt, .wpd, .ps, .html, .hwp, .rtf, .txt	собственный алгоритм, сравнивающий представленные документы с несколькими базами данных (сканирует собственные базы	только для вузов	да

		данных, а также имеет лицензионные соглашения с крупными собственными академическими базами данных)		

ЗАКЛЮЧЕНИЕ

В процессе исследования было проанализировано 27 инструментов для обнаружения текстовых заимствований.

Системы выявления плагиата не разбираются, откуда был заимствован текст – у себя или другого автора. В случае, если работа уже была опубликована, она будет расценена системой как плагиат. Однако неверно считать плагиатом собственный текст автора. Плагиат в отличие от самоплагиата – выставление чужих идей, работ и исследований под своим именем, и с юридической точки зрения плагиат – это преступление против интеллектуальной собственности [4].

Онлайн-сервисы показали лучшую результативность и эффективность в обнаружении заимствований в отличие от программ для персонального компьютера. Тесты проводились на русскоязычных и англоязычных текстах, имеющих различные источники, длину, формат и содержание.

Программам для обнаружения текстовых заимствований свойственны следующие недостатки:

- многие сервисы не поддерживают формат .pdf, что является минусом, так как это один из самых популярных текстовых форматов;
- ряд инструментов не способен проанализировать русскоязычные тексты;
- некоторые системы используют неэффективные алгоритмы и неполные базы данных текстов;
- системы обрабатывают только тексты и имеют пробелы в распознавании математических формул и изображений.

Экспериментальным путем выявлено, что для анализа заимствования текста научных статей наиболее подходящими инструментами со свободным доступом являются онлайн-сервисы Copyleaks, Content Watch и Plagiarisma.NET.

Рекомендуется использовать данные онлайн-системы, так как остальные сервисы и все программы для ПК показали наименьшую эффективность.

СПИСОК ЛИТЕРАТУРЫ

1. *Цвык В.А., Саввина О.В.* Этика науки и этика научных публикаций // URL: http://e-notabene.ru/ca/article_19609.html (дата обращения: 28.04.2019).

2. *Вольное сетевое сообщество «Диссернет».* Итоговый документ конференции «Проблемы качества научной работы и академический плагиат» // URL: https://www.dissernet.org/publications/ivgi_rggu_26.09.2018_itog.htm (дата обращения: 28.04.2019).

3. *Котляров И.Д.* Самоплагиат в научных публикациях // Научная периодика: проблемы и решения, 2011. С. 6–12.

4. *Долотов Р.* Юридическая ответственность за плагиат в научных работах // URL: https://trv-science.ru/2009/12/22/yuridicheskaya-otvetstvennost-za-plagiat-v-nauchnyx-rabotax/ (дата обращения: 28.04.2019).

5. *IEEE* // URL: https://www.ieee.org/publications/rights/plagiarism/plagiarism-faq.html (дата обращения: 28.04.2019).

6. *IEEE* // URL: https://www.ieee.org/publications/rights/plagiarism/ id-plagiarism. html (дата обращения: 28.04.2019).

7. *Шинкаренко В.И., Куропятник Е.С.* Проблемы выявления плагиата и анализ инструментального программного обеспечения для их решения // Наука та прогрес транспорту. Вісник Дніпропетровського національного університету залізничного транспорту, 2017, № 1 (67). С. 133–138.

8. *Luparenko L.* Plagiarism Detection Tools for Scientific e-Journals Publishing // V. Ermolayev et al. (Eds.): ICTERI 2014, CCIS 469, 2014. C. 366–368.

9. *Lin Li, Linlong Xiao, Wenzhen Jin, Hong Zhu, Guocai Yang* Text classification based on Word2vec and Convolutional Neural Network // L. Cheng et al. (Eds.): ICONIP 2018, LNCS 11305, 2018. C.452.

10. *eTXT* // URL: https://www.etxt.ru/antiplagiat/ (дата обращения: 28.04.2019).

11. *Advego Plagiatus* // URL: http://advego.ru/plagiatus/ (дата обращения: 28.04.2019).

12. *Double Content Finder* // URL: https://textbroker.ru/main/dcfinder.html (дата обращения: 28.04.2019).

13. *Viper* // URL: http://www.scanmyessay.com/ (дата обращения: 28.04.2019).

14. *Плагиата.НЕТ* // URL: http://www.mywebs.ru/plagiatanet.html (дата обращения: 28.04.2019).

15. *Plagiarism Checker X* // URL: https://plagiarism-checker-x.en.softonic.com.

16. *Dupli Checker* // URL: http://www.duplichecker.com/ (дата обращения: 28.04.2019).

17. *PaperRater* // URL: https://www.paperrater.com/plagiarism_checker (дата обращения: 28.04.2019).

18. *Plagiarisma.NET* // URL: http://plagiarisma.net (дата обращения: 28.04.2019).

19. *Plagiarism Checker* // URL: http://www.plagiarismchecker.com/ (дата обращения: 28.04.2019).

20. *Plagium* // URL: http://www.plagium.com/ (дата обращения: 28.04.2019).

21. *PlagTracker* // URL: http://www.plagtracker.com/ (дата обращения: 28.04.2019).

22. *PlagScan* // URL: http://www.plagscan.com/ (дата обращения: 28.04.2019).

23. *Plagiarism Detector* // URL: http://plagiarismdetector.net/ (дата обращения: 28.04.2019).

24. *Content Watch* // URL: http://www.content-watch.ru/text/ (дата обращения: 28.04.2019).

25. *Grammarly* // URL: http://www.grammarly.com/ (дата обращения: 28.04.2019).

26. *Docoloc* // URL: https://www.docoloc.de/ (дата обращения: 28.04.2019).

27. *Text.ru* Онлайн-сервис проверки текста на уникальность // URL http://text.ru/ (дата обращения: 28.04.2019).

28. *Антиплагиат* // URL: http://www.antiplagiat.ru/ (дата обращения: 28.04.2019).

29. *eTXT* Проверка текста на уникальность онлайн // URL: https://www.etxt.ru/antiplagiat/ (дата обращения: 28.04.2019).

30. *Turnitin* // URL: https://www.turnitin.com (дата обращения: 28.04.2019).

31. *IThenticate* // URL: http://www.ithenticate.com (дата обращения: 28.04.2019).

32. *Quetext* // URL: https://www.quetext.com (дата обращения: 28.04.2019).

33. *Copyleaks* // URL: https://copyleaks.com (дата обращения: 28.04.2019).

34. *Scribbr* // URL: https://www.scribbr.com (дата обращения: 28.04.2019).

35. *Strikeplagiarism* // URL: https://strikeplagiarism.com/ (дата обращения: 28.04.2019).

36. *Small SEO* // URL: https://smallseotoolz.net (дата обращения: 28.04.2019).

REVIEW OF EXISTING TOOLS FOR DETECTING PLAGIARISM AND SELF-PLAGIARISM

A. E. Tlitova[1], A. S. Toschev [2]

[1,2]*GDC Services*

[1]soloadelante15@gmail.com, [2]atoschev@kpfu.ru

Abstract

All the time scientists need to publish the results of their work in order to remain relevant, meet the time, criteria, and not be outside the scientific community. The well-known principle of “publish or perish” often forces scientists to strive for quantity, not quality [1]. Along with the problems of authorship, paid research, the fabrication of the results, plagiarism and self-plagiarism are among the most common violations. Their impact is more subtle, but no less disruptive for the scientific community.

The article provides an overview of the existing tools for identifying borrowing in the scientific articles of the authors. Decisions’ analysis is performed by com-paring systems for a number of characteristics. The tools are tested on real data to investigate their performance and efficiency.

Keywords*: Plagiarism, self-plagiarism, scientific ethics, text borrowing, text analysis*

REFERENCES

1. *Tsvyk V.A., Savvina O.V.* Etika nauki i etika nauchnyh publikacij. http://e-notabene.ru/ca/article_19609.html (2017).

2. *Volnoe setevoe soobshchestvo «Dissernet»* Itogovyj dokument konferencii «Problemy kachestva nauchnoj raboty i akademicheskij plagiat». https://www.dissernet.org/publications/ivgi_rggu_26.09.2018_itog.htm. (2018)

3. *Kotlyarov I.D.* Samoplagiat v nauchnyh publikaciyah. Nauchnaya periodika: problemy i resheniyapp. pp. 6–12 (2011).

4. *Dolotov R.* Yuridicheskaya otvetstvennost za plagiat v nauchnyh rabotah. https://trv-science.ru/2009/12/22/yuridicheskaya-otvetstvennost-za-plagiat-v-nauchnyx-rabotax/ (2009).

5. *Luparenko L.* Plagiarism Detection Tools for Scientific e-Journals Publishing, V. Ermolayev et al. (Eds.): ICTERI 2014, CCIS 469. pp.366–368 (2014).

6. *Shinkarenko V.I., Kuropyatnik E.S.* Problems of Plagiarism Detection and Analysis of Tool Software for Solving Them, Nauka ta Progress Transport. Bulletin of Dnipropetrovsk National University of Social and Transport, No. 1 (67). pp. 133–138 (2017).

7. *Lin Li, Linlong Xiao, Wenzhen Jin, Hong Zhu, Guocai Yang* Text selection based on Word2vec and Convolutional Neural Network. L.Cheng et al. (Eds.): ICONIP 2018, LNCS 11305. pp. 452 (2018).

8. *Advego Plagiatus*. http://advego.ru/plagiatus/, last accessed 2019/04/28.

9. *Double Content Finder*. https://textbroker.ru/main/dcfinder.html, last accessed 2019/04/28.

10. *eTXT*. https://www.etxt.ru/antiplagiat/, last accessed 2019/04/28.

11. *Plagiarism.NET*. http://www.mywebs.ru/plagiatanet.html, last accessed 2019/04/28.

12. *Plagiarism Checker X*. https://plagiarism-checker-x.en.softonic.com, last accessed 2019/04/28.

13. *Viper*. http://www.scanmyessay.com/, last accessed 2019/04/28.

14. *Antiplagiat*. http://www.antiplagiat.ru/, last accessed 2019/04/28.

15. *Content Watch*. http://www.content-watch.ru/text/, last accessed 2019/04/28.

16. *Copyleaks*. https://copyleaks.com, last accessed 2019/04/28.

17. *Docoloc*. https://www.docoloc.de/, last accessed 2019/04/28.

18. *Dupli Checker*. http://www.duplichecker.com/, last accessed 2019/04/28.

19. *Grammarly*. http://www.grammarly.com/, last accessed 2019/04/28.

20. *IEEE.* https://www.ieee.org/publications/rights/plagiarism/id-plagiarism.html, last accessed 2019/04/28.

21. *IEEE FAQ.* https://www.ieee.org/publications/rights/plagiarism/plagiarism-faq.html, last ac-cessed 2019/04/28.

22. *IThenticate*. http://www.ithenticate.com, last accessed 2019/04/28 [20].

23. *PaperRater*. https://www.paperrater.com/plagiarism_checker, last accessed 2019/04/28.

24. *Plagiarisma.NET*. http://plagiarisma.net, last accessed 2019/04/28.

25. *Plagiarism Checker*. http://www.plagiarismchecker.com/, last accessed 2019/04/28.

26. *Plagium*. http://www.plagium.com/, last accessed 2019/04/28.

27. *PlagTracker*. http://www.plagtracker.com/, last accessed 2019/04/28.

28. *PlagScan*. http://www.plagscan.com/, last accessed 2019/04/28.

29. *Plagiarism Detector*. http://plagiarismdetector.net/, last accessed 2019/04/28.

30. *Quetext*. https://www.quetext.com, last accessed 2019/04/28.

31. *Scribbr*. https://www.scribbr.com, last accessed 2019/04/28.

32. *Small SEO Tools*. https://smallseotoolz.net, last accessed 2019/04/28.

33. *Strikeplagiarism*. https://strikeplagiarism.com/, last accessed 2019/04/28.

34. *Text.ru*. Online service for checking the text for uniqueness. http://text.ru/, last accessed 2019/04/28.

35. *Turnitin*. https://www.turnitin.com, last accessed 2019/04/28.

СВЕДЕНИЯ ОБ АВТОРАХ

ТЛИТОВА Алина Эдуардовна – магистр Высшей школы ИТИС, инженер по тестированию программного обеспечения.

Alina Eduardovna TLITOVA – Master of ITIS Higher School, Software Testing Engineer

email: soloadelante15@gmail.com

ТОЩЕВ Александр Сергеевич – ассистент кафедры программной инженерии, инженер-проектировщик программного обеспечения.

Alexander Sergeevich TOSHCHEV – Assistant of Software Engineering Department, Software Design Engineer.

email: atoschev@kpfu.ru

Материал поступил в редакцию 21 июня 2019 года

УДК 004.514+004.42

АВТОМАТИЗАЦИИ ПРОЦЕССА РАЗРАБОТКИ ИНТЕРАКТИВНЫХ ПРОТОТИПОВ ANDROID-ПРИЛОЖЕНИЙ НА ОСНОВЕ НИЗКОДЕТАЛИЗИРОВАННЫХ МАКЕТОВ

А. С. Хлопунов[1], И. С. Шахова[2]

[1,2]Высшая школа информационных технологий и интеллектуальных систем Казанского (Приволжского) федерального университета

[1]hlopunov598@gmail.com, [2]is@it.kfu.ru

Аннотация

Приведены механизмы автоматизации процесса разработки интерактивных прототипов мобильных приложений на основе рукописных макетов. Процесс автоматизации включает в себя использование методов машинного обучения для распознавания рукописных макетов. Для обеспечения взаимодействия пользователя с предложенными механизмами реализовано мобильное Android-приложение.

***Ключевые слова**: прототипирование, UI, UX, мобильные приложения, пользовательский интерфейс*

ВВЕДЕНИЕ

Процесс разработки приложения состоит из нескольких этапов, и от продолжительности каждого из них зависят конечная стоимость продукта и время его разработки. Основные из этих этапов:

1) поиск идеи и анализ альтернативных решений;
2) сбор требований и составление технического задания;
3) разработка прототипа приложения и согласование его с заказчиком;
4) разработка дизайна приложения и реализация функционала;
5) тестирование и согласование с заказчиком [1].

По данным, опубликованным в статье [2], более 50% опрошенных разработчиков считают, что проектирование прототипов экранов является

первоначальной задачей. Это позволяет в кратчайшие сроки получить обратную связь от целевой аудитории и тем самым выявить проблемы в макете на начальном этапе и снизить время и стоимость разработки [3].

Прототипы могут быть как низкодетализированными (нарисованными на бумаге или с помощью специальных инструментов), так и полностью интерактивными, обладающими высокой детализацией и позволяющими взаимодействовать с ними. Соответственно, чем детальнее прототип, тем более точная обратная связь может быть по нему получена. Такие проработанные прототипы позволяют взаимодействовать с экранами макетов приложения: нажимать кнопки, вводить текст, тем самым давая больше отзывов о работе приложения [4]. Однако создание интерактивного прототипа представляет собой более трудозатратный процесс в сравнении с созданием низкодетализированных макетов, а также предполагает наличие навыков работы с инструментами прототипирования [5, 6].

Таким образом, целью данной работы является разработка инструмента автоматизации процесса создания интерактивного прототипа на основе рукописных макетов экранов приложения для операционной системы Android.

ОБЗОР СУЩЕСТВУЮЩИХ РЕШЕНИЙ

При обзоре альтернативных решений для генерации элементов интерфейса из рукописных макетов были обнаружены следующие инструменты:

- Система проектирования Airbnb. Данная система позволяет распознавать рукописные макеты экранов и динамически генерировать из них код. На демонстрационных роликах ИИ данной системы показывает хорошие результаты распознавания различного типа элементов экрана. Однако данный инструмент предназначен для веб-приложений и не подходит для мобильных, а также на данный момент доступен только внутри корпоративной сети Airbnb.
- Приложение Prott. С помощью данного приложения можно сфотографировать нарисованные прототипы экранов, пометить переходы для кнопок, а затем загрузить все это в графический редактор Sketch, где будет сгенерирован интерактивный прототип. Однако данный программный инструмент не решает поставленную

проблему в полной мере, так как требует взаимодействия со сторонними инструментами.

РАСПОЗНАВАЕМЫЕ ЭЛЕМЕНТЫ ИНТЕРФЕЙСА

Для прототипирования и обучения нейронной сети был определен следующий набор элементов, используемых в дизайне мобильных приложений. Данный выбор был обусловлен тем, что часть этих элементов является основным компонентом дизайна приложений на Android, а другие являются часто используемыми. Для каждого из них был определен свой стандартный внешний вид.

Были определены следующие элементы интерфейса:

- Button. Кнопка, которая состоит из текста или картинки и обрабатывает касания пользователей (рис. 1).

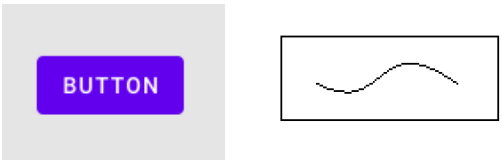


Рис. 1. Button в приложении и вид, заданный в рукописном макете

- ImageView. Данный компонент является контейнером для отображения изображений (рис. 2).

Рис. 2. ImageView в приложении и вид, заданный в рукописном макете

- Spinner. Выпадающий список, который предоставляет быстрый способ выбора одного значения из набора. В свернутом состоянии показывает текущее выбранное значение, а при касании отображается раскрывающееся меню со всеми остальными значениями (рис. 3).

Рис. 3. Spinner в приложении и вид, заданный в рукописном макете

- EditText. Элемент пользовательского интерфейса для ввода и изменения текста (рис. 4).

Рис. 4. EditText в приложении и вид, заданный в рукописном макете

- TextView. Элемент для отображения текста (рис. 5).

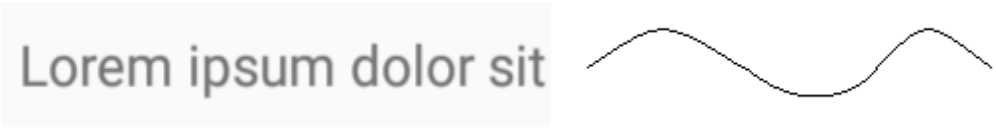


Рис. 5. TextView в приложении и вид, заданный в рукописном макете

- CheckBox. Данный компонент является флагом, с помощью которого можно выбрать нужные опции из определенного набора значений (рис. 6).

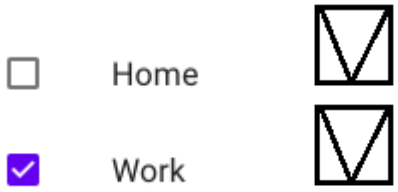


Рис. 6. CheckBox в приложении и вид, заданный в рукописном макете

- RadioButton. Элемент, который предоставляет выбрать только один пункт из предопределенного набора значений. Никогда не используется в одиночестве, всегда есть несколько таких элементов, объединенных в одну группу (рис. 7).

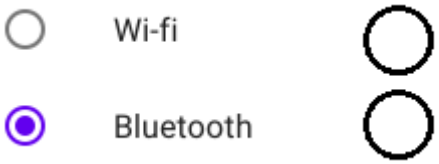


Рис. 7. RadioButton в приложении и вид, заданный в рукописном макете

- Switch. Компонент представляет собой полосу с двумя состояниями. Он используется, когда необходимо сделать выбор между двумя противоположными вариантами (рис. 8).

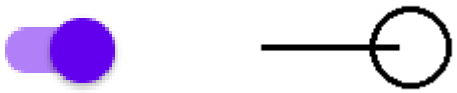


Рис. 8. Switch в приложении и вид, заданный в рукописном макете

- Floating Action Button. Плавающая кнопка, которая обычно выполняет основное или самое распространенное действие на экране. Отображается перед всем содержимым экрана (рис. 9).

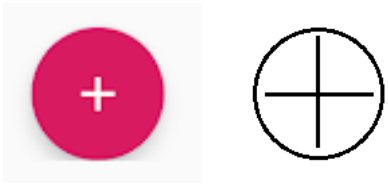


Рис. 9. Floating Action Button приложение и вид, заданный в рукописном макете

АЛГОРИТМ РАСПОЗНАВАНИЯ И ГЕНЕРАЦИИ ЭЛЕМЕНТОВ

Алгоритм распознавания элементов с фотографий нарисованных макетов и последующей генерации экранов на их основе использует предварительно подготовленную и обученную нейронную сеть. Для ее обучения использовалось 150 рукописных шаблонов и 2000 сгенерированных с помощью скрипта. Данный алгоритм состоит из следующих основных шагов (рис. 10):

1) добавление фотографии в приложение и сохранение пути до нее;
2) создание классификатора для распознавания изображений. На вход подаются модель нейронной сети, файл с метками элементов, которые могут быть распознаны, и ожидаемые размеры изображений после сжатия;
3) первый запуск метода decodeResource() для Bitmap с флагом inJustDecodeBounds = true, что позволяет узнать размеры изображения без его загрузки в оперативную память;
4) вычисление коэффициента сжатия;
5) второй запуск метода decodeResource() со сжатым изображением;
6) запуск распознавания на Bitmap, полученном из изображения;
7) заполнение листа элементами, генерируемыми из параметров, полученных посредством распознавания: идентификатора, метки (названия элемента), точности распознавания, координат прямоугольной области, в которой находится элемент;
8) вычисление метрик экрана и коэффициентов отношений высоты и ширины между устройством и сжатым изображением;
9) парсинг пришедших результатов и генерация элементов интерфейса в зависимости от метки;
10) вычисление расположения и размеров элементов на основе координат прямоугольной области и коэффициентов отношений ширины и высоты;
11) генерация из полученных элементов экрана.

Запуск нейронной сети и процесс распознавания элементов происходит только один раз для каждого изображения. После процесса распознавания полученные из него элементы, удовлетворяющие по точности распознавания алгоритму генерации (в данном случае 65%), записываются в таблицу в базе данных. При последующих запусках элементы извлекаются из базы данных, конвертируются в соответствующий формат и подаются на вход алгоритму генерации экрана.

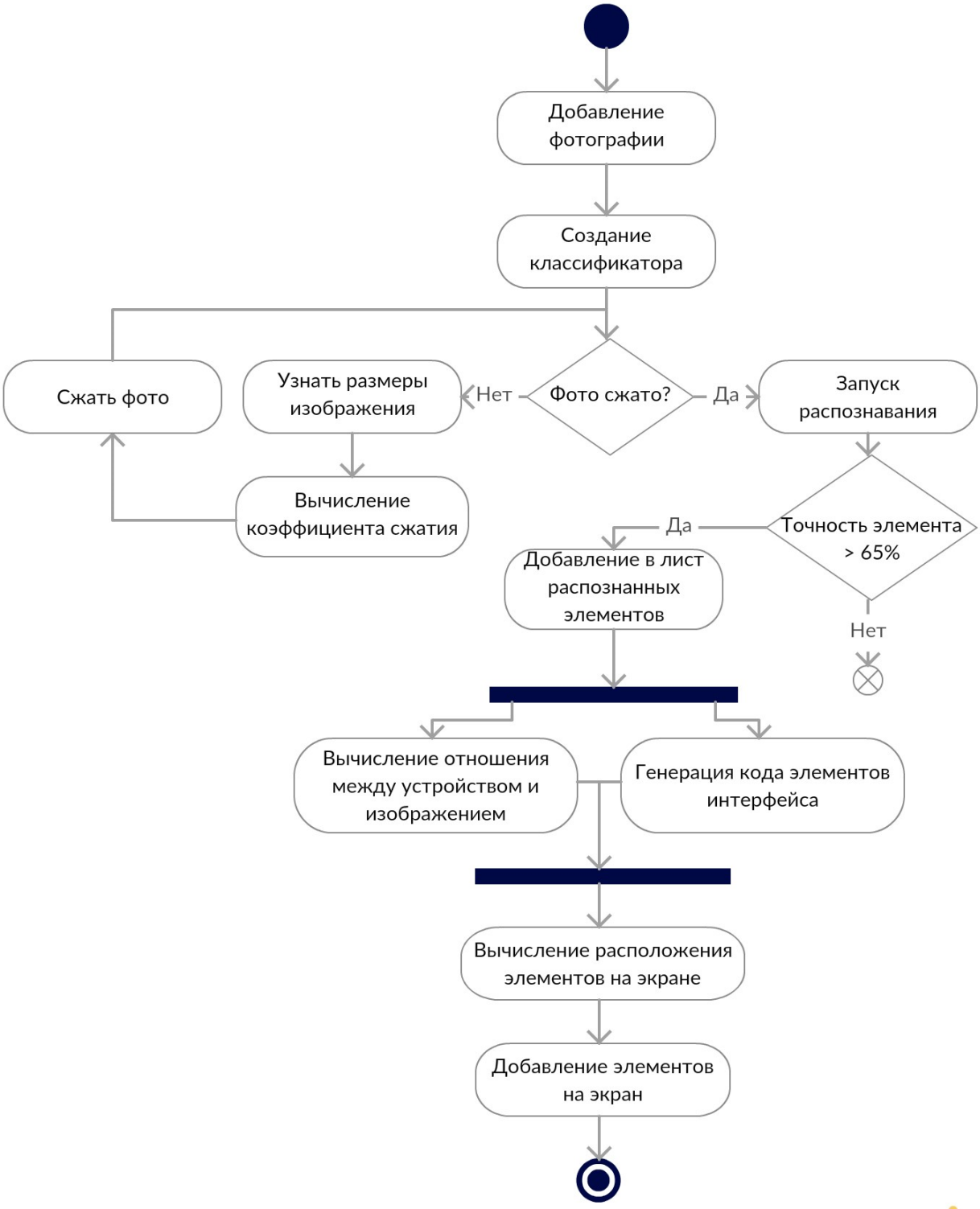


Рис. 10. Алгоритм распознавания и генерации элементов

ОПИСАНИЕ РАБОТЫ ПРИЛОЖЕНИЯ

Работа пользователя с разработанным инструментом осуществляется через графический интерфейс мобильного приложения. На рисунке 11 изображена диаграмма сценариев использования мобильного приложения.

Рис. 11. Диаграмма сценариев использования приложения

После запуска приложения и выдачи ему разрешений на доступ к файлам на устройстве пользователю предлагается создать новый проект – это область, где хранятся фотографии макетов экранов для каждого отдельного заказчика или проекта. После выбора проекта открывается галерея приложения, в которой хранятся загруженные шаблоны. Здесь же можно загрузить и новые фотографии с устройства (рис. 12). При клике по шаблону его изображение передается в нейронную сеть и открывается сгенерированное на его основе окно.

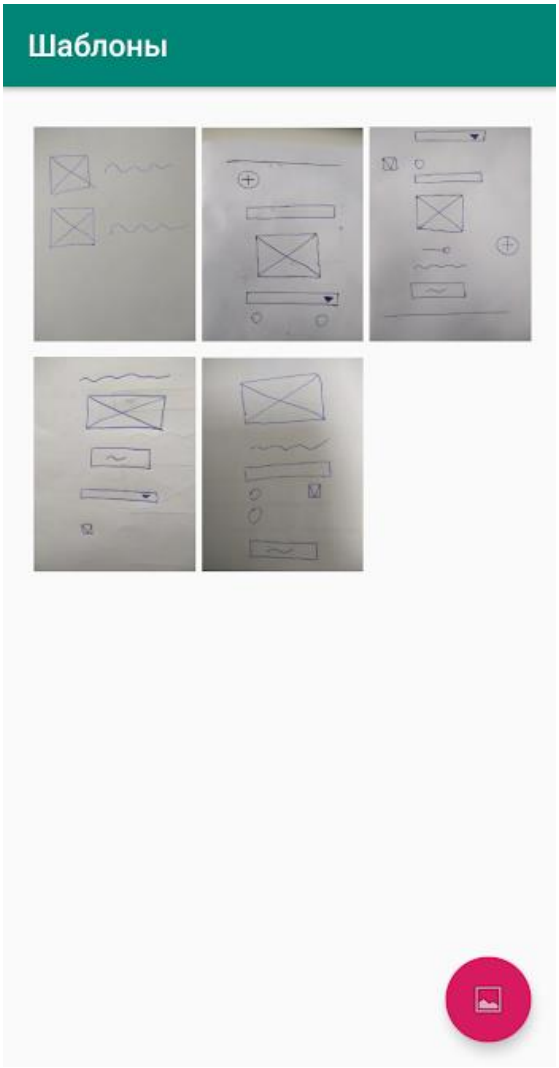


Рис. 12. Экран галереи приложения

Все элементы являются интерактивными, то есть в поля для ввода текста можно вводить текст, в выпадающих списках можно выбрать одно из значений и др. Также есть возможность добавления перехода с одного экрана на другой через нажатие обычной кнопки или плавающей кнопки. Для этого необходимо удерживать кнопку в течение 1–2 секунд, и после этого будет предложено выбрать экран из этого же проекта, на который необходимо производить переход. В следующий раз при клике на эту кнопку произойдет переход на выбранный экран. Пример генерации интерактивного экрана по низкодетализированному шаблону изображен на рисунке 13.

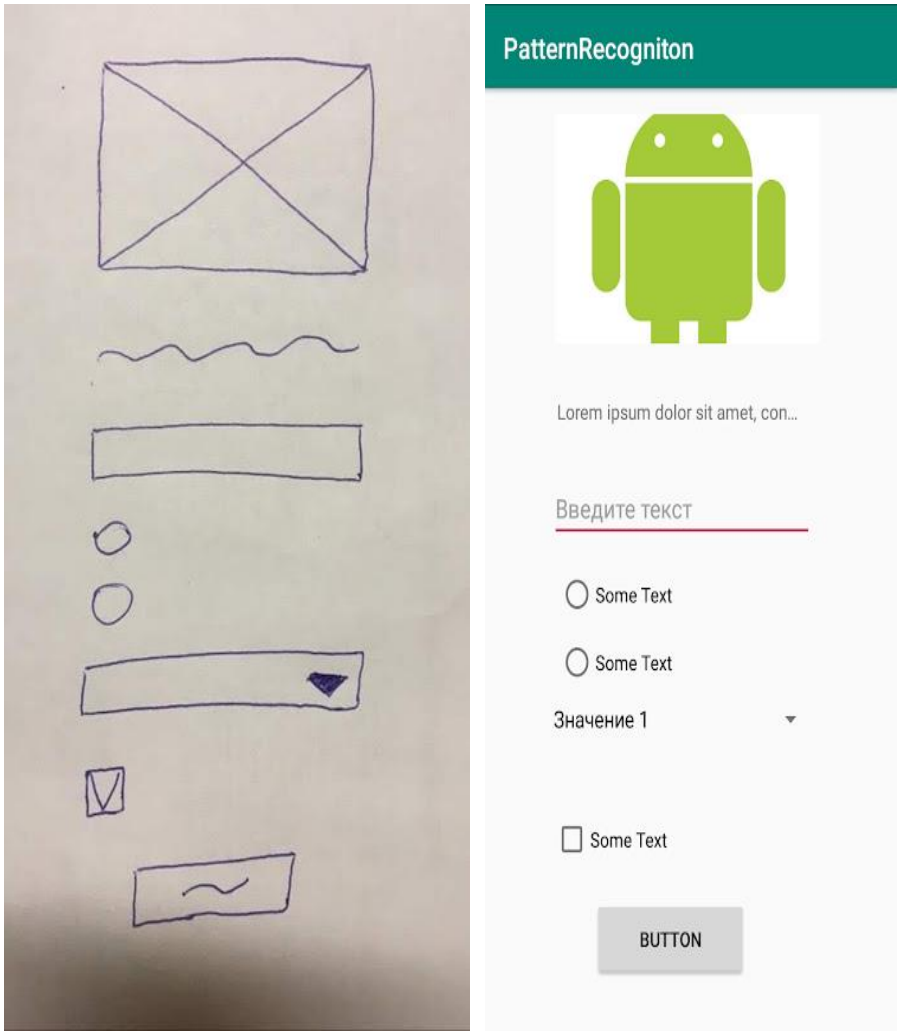


Рис. 13. Низкодетализированный прототип и сгенерированный на его основе экран

ЗАКЛЮЧЕНИЕ

Описан разработанный инструмент для автоматической генерации интерактивных прототипов на основе низкодетализированных макетов для устройств под управлением операционной системы Android. Разработанное мобильное приложение дает пользователю возможность генерации интерактивного интерфейса приложения по нарисованному шаблону.

Разработанный инструмент позволяет сократить временные затраты на создание интерактивного прототипа, тем самым обеспечивая получение детальной обратной связи по прототипу на начальном этапе разработки.

СПИСОК ЛИТЕРАТУРЫ

1. *Vithanti T., Kumar A.* Modeling the Mobile Application Development Lifecycle // International MultiConference of Engineers and Computer Scientists. 2014. V. 1. P. 596-600.

2. *Harleen K.F., Xiaofeng W., Swati V.C.* An Investigation into Mobile Application Development Processes: Challenges and Best Practices // I.J. Modern Education and Computer Science. 2014. V. 6. P. 1–9.

3. *Why Prototyping is Essential to Your Design Process.* URL: https://www.webfx.com/blog/web-design/ prototyping-is-essential/.

4. *Kamushken R.* The advantages of interactive prototyping // Medium. 2017. URL: https://medium.com/@kamushken/the-advantages-of-interactive-prototyping-855203728b83.

5. *Babich N.* The Magic of Paper Prototyping // UX Planet. 2018. URL: https://uxplanet.org/the-magic-of-paper-prototyping- 51693eac6bc3

6. *Morson S.* Using Wireframes to Design Your App // Learn Design for iOS Development. 2014. P. 69–84.

AUTOMATION OF ANDROID APPLICATIONS INTERACTIVE PROTOTYPES DEVELOPMENT BASED ON LOW-FIDELITY WIREFRAMES

A. S. Khlopunov[1], I. S. Shakhova[2]

Higher School of Information Technologies and Intelligent Systems at Kazan Federal University

[1]hlopunov598@gmail.com, [2]is@it.kfu.ru

Abstract

Some mechanisms for automation of Android applications interactive prototypes development based on handwritten wireframes are described in the paper. The process of automation includes machine learning methods used for the handwritten wireframes recognition. The mobile Android application is developed to ensure user interaction with these mechanisms.

Keywords*: prototyping, UI, UX, mobile applications, user interface*

REFERENCES

1. *Vithanti T., Kumar A.* Modeling the Mobile Application Development Lifecycle // International MultiConference of Engineers and Computer Scientists. 2014. V. 1. P. 596–600.

2. *Harleen K.F., Xiaofeng W., Swati V.C.* An Investigation into Mobile Application Development Processes: Challenges and Best Practices // I.J. Modern Education and Computer Science. 2014. V. 6. P. 1–9.

3. *Why Prototyping is Essential to Your Design Process.* URL: https://www.webfx.com/blog/web-design/ prototyping-is-essential/.

4. *Kamushken R.* The advantages of interactive prototyping // Medium. 2017. URL: https://medium.com/@kamushken/the-advantages-of-interactive-prototyping-855203728b83.

5. *Babich N.* The Magic of Paper Prototyping // UX Planet. 2018. URL: https://uxplanet.org/the-magic-of-paper-prototyping- 51693eac6bc3

6. *Morson S.* Using Wireframes to Design Your App // Learn Design for iOS Development. 2014. P. 69–84

СВЕДЕНИЯ ОБ АВТОРАХ

ХЛОПУНОВ Анатолий Сергеевич **–** студент Высшей школы информационных технологий и интеллектуальных систем Казанского федерального университета, Android-разработчик**.**

Anatoly Sergeevich KHLOPUNOV – student of the Higher School ofInformation Technologies and Intelligent Systems of Kazan Federal University, developer of software for Android.

email: hlopunov598@gmail.com

ШАХОВА Ирина Сергеевна – ассистент кафедры программной инженерии Высшей школы информационных технологий и интеллектуальных систем Казанского федерального университета. Сфера научных интересов – мобильные приложения, цифровые образовательные системы, индивидуализация образования, мобильное обучение.

Irina Sergeevna SHAKHOVA – teacher of the Higher School of Information Technologies and Intelligent Systems, Kazan Federal University. Research interests include mobile applications, digital educational systems, individualization in education, mobile learning.

email: is@it.kfu.ru

Материал поступил в редакцию 23 июня 2019 года

УДК 004.622+004.658.6

СОЗДАНИЕ МЕТОДА СРАВНЕНИЯ РЕЛЯЦИОННЫХ ТАБЛИЦ

А. Ш. Якупов[1], Д. А. Клинов[2]

[1]ООО «Пьяно Тек»; [2]ООО «Делион»

[1]asyakupov@kpfu.ru, [2]daniil.klinov@delion.ru

Аннотация

Статья посвящена созданию быстрого метода сравнения огромного количества данных таблиц в рамках реляционных систем управления базами данных. Проведено исследование существующих решений и показана востребованность создания эффективного метода сравнения реляционных отношений. Создан алгоритм с использованием вероятностной структуры данных «Исчисляемый фильтр Блума» и метода Монте-Карло. Предлагаемое решение уникально в своем направлении, так как использует наименьшее количество временных ресурсов. Построена вероятностная модель созданного алгоритма. В процессе написания статьи были выявлены пути развития алгоритма в сторону внедрения параллелизации процессов.

Ключевые слова: *мультимножество, сравнение реляционных таблиц, гетерогенная система, исчисляемый фильтр Блума, метод Монте-Карло, репликация, Oracle, PostgreSQL, вероятностная структура данных*

ВВЕДЕНИЕ

В современном мире наблюдается рост количества информации. Согласно статистике аналитической фирмы IDC «Эра данных 2025» [13], объем данных, которые человечество накопит уже меньше чем через 10 лет, составит 163 зеттабайт. Для сравнения: весь мировой объем интернет-трафика в 2016 году едва превысил 1 зеттабайт.

Для хранения огромного количества данных требуются мощные и современные системы управления базами данных (СУБД), примерами которых являются Oracle, PostgreSQL, MySQL, Microsoft SQL Server, MongoDB и другие [12]. На сегодняшний день наблюдается рост популярности использования

PostgreSQL [12]. В России это обосновано развитием сообщества благодаря мероприятиям PG Day и PG Conf и постоянным расширением функциональных возможностей PostgreSQL. Массовая миграция данных в рамках импортозамещения поднимает вопрос сравнения перенесенных данных.

При переносе большого количества данных путем замены одной СУБД на другую специалисты не могут гарантировать корректность и целостность переноса в связи с тем, что могут возникнуть внутренние, непредсказуемые ошибки из-за различий реализаций двух реляционных баз данных, также ошибками, связанными с различиями типов данных, внутренних функций, синтаксиса последовательностей и так далее [14]. Процесс миграции нужно рассматривать в комплексе с обеспечением мер по отказоустойчивости, резервированию и безопасности новой системы. Существующие на данный момент инструменты предоставляют возможность сравнения только построчно [6–10]. Данный подход является большим неудобством в мире, когда данные превышают несколько сотен гигабайт на одну реляционную таблицу из-за ограничений временных ресурсов.

СОПОСТАВЛЕНИЕ И АНАЛИЗ СУЩЕСТВУЮЩИХ ИНСТРУМЕНТОВ И РЕШЕНИЙ ДЛЯ СРАВНЕНИЯ ТАБЛИЦ РСУБД

Рассмотрим особенности существующих решений:

- **Red Gate SQL Data Compare**. Возможность копирования и переноса данных поиска из баз данных разработки в стадию производства. Генерация сценариев T-SQL для обновления одной базы данных с содержимым другой. Возможность вести точную историю всех предыдущих записей баз данных. Возможность сравнения и синхронизации данных, хранящихся в SQL Server Management Studio.
- **dbForge Data Compare**. Ручная настройка методов сравнения, позволяющая сократить время простоя системы, вызванное ошибками репликации. данных, и ускорение восстановления. Ускоренная разработка приложений, благодаря быстрому внедрению изменений данных. Индивидуальные сценарии синхронизации данных при низкой стоимости. Большая эффективность при сравнении больших баз данных. Широкая поддержка версий SQL Server.

- **EMS Data Comparer.** Отсутствие инструментов по сравнению структуры базы данных. Русификация от разработчика. Визуальное представление различий между данными в базах. Автоматическая и ручная выборки данных для сравнения. Возможность сохранять синхронизирующий сценарий. Большое разнообразие параметров для сравнения и синхронизации данных. Автоматическое сравнение/синхронизация данных. Возможность сохранения всех параметров, заданных в активной сессии. Гибкий графический интерфейс
- **SQLDelta**. Встроенное управление индексами, ключами и зависимостями между таблицами. Отчет в виде подробного html-файла. Загрузка баз данных происходит асинхронно.
- **SQL Comparison toolset from Idera**. Ведение историй сеансов сравнения. Обширный выбор фильтров, сортировок для системной аналитики после сравнения. Настройка сравнения, учитывающая ключи, столбцы, индексы. Не требует установки компонентов, динамических библиотек и ссылок.

Были изучены и проанализированы 5 инструментов сравнения таблиц РСУБД, которые суммарно имеют более 400 тысяч пользователей [6–10]. Все изученные инструменты используют построчный алгоритм сравнения таблиц. Некоторые инструменты используют параллелизацию алгоритма сравнения таблиц, но данный подход не является непосредственным преимуществом построчного сравнения таблиц перед сравнением таблиц с помощью сравнения их Исчисляемых фильтров Блума, так как параллелизацию можно использовать и при построении Исчисляемых фильтров Блума на таблицу.

После детального анализа существующих инструментов и решений для сравнения таблиц РСУБД не удалось найти инструмент, который бы использовал какую-либо иную технологию сравнения таблиц от сравнения таблиц построчно.

ОПИСАНИЕ АЛГОРИТМА СРАВНЕНИЯ ТАБЛИЦ И ЕГО ЭФФЕКТИВНОСТЬ

Целью конечного алгоритма является выявление простого факта несоответствия двух сравниваемых реляционных отношений с произвольным количеством записей и произвольным количеством атрибутов реляционных баз данных.

Задача состоит из нескольких этапов проверок, которые избегают использования построчного и последующего атрибутного сравнений данных.

В данный момент времени существующие на рынке инструменты сравнения используют обычный алгоритм перебора. Сложность алгоритма перебора составляет O(nm), где n – количество строк реляционного отношения, m – количество столбцов реляционного отношения.

Более того, существующие инструменты очень активно используют ресурсы компьютера/сервера, загружая части или весь объем данных двух сравниваемых отношений в оперативную память. Это связано не только с объемом данных, но и с производимой сортировкой атрибутов сравниваемых отношений. Алгоритм сортировки требует дополнительных затрат, и его сложность в худшем случае составляет O(mn^2), где n – количество строк реляционного отношения, m – количество столбцов реляционного отношения.

В свою очередь для выполнения сравнения Исчисляемых фильтров Блума достаточно их построить и сравнить между собой. Нет никакой сортировки и построчного сравнения каждого значения столбца в строке таблицы.

Сложность алгоритма в худшем случае линейна. Если требуется повторное сравнение таблиц после дополнительной миграции данных, то не требуется строить Исчисляемый фильтр Блума для таблиц, так как он может быть сохранен как заранее приготовленный вектор каждой из таблиц, что даст экономию в ресурсах вычислительной машины.

Этапы сравнения двух реляционных таблиц сводятся к последовательному выполнению шагов:

1. Сравнение типов таблиц;
2. Сравнение базовой статистики двух таблиц;
3. Построение и сравнение Исчисляемых фильтров Блума двух таблиц;
4. Сравнение таблиц с помощью метода Монте-Карло.

ВЕРОЯТНОСТНАЯ МОДЕЛЬ СТРУКТУРЫ ДАННЫХ «ИСЧИСЛЯЕМЫЙ ФИЛЬТР БЛУМА»

Необходимо сделать расчет вероятности равенства двух таблиц, у каждой из которых построен Исчисляемый фильтр Блума. Расчет вероятности равенства двух таблиц осуществляется с помощью сравнений двух заранее построенных Исчисляемых фильтров Блума.

Если при проверке равенства двух множеств с помощью сравнения их Исчисляемых фильтров Блума было установлено, что два множества не равны, то данных ответ является однозначным и не имеет погрешностей, так как Исчисляемые фильтры Блума, построенные на одинаковые множества элементов, не могут различаться.

Если при проверке равенства двух множеств с помощью сравнения их Исчисляемых фильтров Блума было установлено, что два множества равны: либо два множества действительно равны, либо был получен ложный положительный ответ, так как при построении Исчисляемых фильтров Блума двух неравных множеств с помощью случайного распределения данных были получены идентичные значения в каждом из соответствующих значений двух массивов Исчисляемых фильтров Блума.

Необходимо вычислить вероятность события, когда были построены два одинаковых Исчисляемых фильтра Блума для двух неравных множеств элементов.

Точный расчет данной вероятности не позволяет осуществить отсутствие информации о количестве неравных элементов двух множеств, при их наличии. На основании теории вероятностей и математической статистики можно наверняка вывести вероятностную формулу при наличии информации о количестве неравных элементов. Заранее неизвестно, какое количество элементов двух множеств не равны друг другу, но на основании переменной, которая будет определять количество неравных элементов, можно вычислить вероятность равенства двух множеств при сравнении соответствующих Исчисляемых фильтров Блума.

При наличии двух неравных множеств элементов вероятность события, когда были построены идентичные Исчисляемые фильтры Блума для этих

множеств, вычисляется по формуле

$$P = 1 / ((m + x * k - 1)! / ((x * k)! * (m - 1)!)),$$

где *P* – это вероятность события, *m* – количество ячеек в массиве Исчисляемого фильтра Блума, *x* – количество неравных элементов двух множеств элементов, *k* – количество объявленных хэш-функций при построении Исчисляемых фильтров Блума. Очевидно, что данная формула зависит от количества неравных элементов двух множеств. Другими словами, расчетная формула может дать ответ о том, что не установлено неравенство двух множеств с определенной вероятностью на основании сравнения двух Исчисляемых фильтров Блума.

Данная формула определяет количество вариаций выбора *kx* ссылок на определенные значения массива Исчисляемого фильтра Блума, которые образовали равное множество совместно со ссылками неравных элементов другого Исчисляемого фильтра Блума, инициировав тем самым ложноположительный ответ сравнения Исчисляемых фильтров Блума. Данное значение всегда равно 1, так как заранее можно установить, по какому набору ссылок неравных элементов сравниваемых неравных множеств образовались равные множества ссылок. Данное значение однозначное и неизменяемое, поэтому существует только один способ выбора *kx* ссылок на определенные значения массива Исчисляемого фильтра Блума.

Также данная формула определяет количество вариаций выбора *kx* элементов из множества с количеством элементов *m*.

На основании данных показателей вычисляется вероятность того, что произошло построение равных Исчисляемых фильтров Блума для неравных множеств элементов. Вычтя данную вероятность из единицы, получим вероятность того события, что произошло построение равных Исчисляемых фильтров Блума для равных множеств элементов.

Данная формула основывается на принципах теории вероятностей и комбинаторики, а именно, вычислении вероятности события и сочетания с повторениями.

На основании полученного значения по данной формуле определяется вероятность ложноположительного ответа, а на основании

ложноположительного ответа вычисляется вероятность верного положительного ответа, с учетом значения переменных *m* и *x* в самом худшем случае, а именно, когда *x* принимает значение 1.

ЗАКЛЮЧЕНИЕ

Проанализированы существующие инструменты сравнения реляционных таблиц и выявлены их недостатки. Разработано решение с использование вероятностной структуры данных «Исчисляемый фильтр Блума» и метода Монте-Карло для эффективного сравнения таблиц РСУБД.

Доказана эффективность данного алгоритма сравнения таблиц в разделе «Описание алгоритма сравнения таблиц и его эффективность» настоящей статьи и построена вероятностная модель алгоритма. Продолжить развитие приведенного алгоритма для последующей научной деятельности можно в сторону параллелизации процессов.

СПИСОК ЛИТЕРАТУРЫ

1. *Birialtsev E.* Intelligent search in Big Data [Text] // Approach to Data Integration. 2017. V. 46. No 19. P. 7–14.

2. *Chen G., Guo D., Luo L., Ren B.* Optimization of multicast source routing based on bloom filter // IEEE Communication Letters. 2018. No 4. P. 700–703.

3. *Kareev I.* Lower bounds for expected sample size of sequential procedures for the multinomial selection problems // Communications in Statistics. 2017. V. 913. No 1. P. 1–29.

4. *Wu K., Tan H., Liu Y., Zhang J., Zhang Q., Ni L.* Side channel: Bits over interference // IEEE Transactions on Mobile Computing. 2017. No 8. P. 1317–1330.

5. *Афанасьев Г.И., Марков А.Д.* База Данных NoSql и их сравнение с традиционными базами данных // Теория Инноваций. 2017. № 5-2. С. 4–10.

6. Официальная документация к инструменту сравнения таблиц РСУБД “Devart” [Электронный ресурс]. Режим доступа: https://www.devart.com (Дата обращения: 19.11.2018).

7. Официальная документация к инструменту сравнения таблиц РСУБД “Idera” [Электронный ресурс]. Режим доступа: https://www.idera.com (Дата обращения: 17.01.2019).

8. Официальная документация к инструменту сравнения таблиц РСУБД “Red Gate” [Электронный ресурс]. Режим доступа: https://www.red-gate.com (Дата обращения: 11.11.2018).

9. Официальная документация к инструменту сравнения таблиц РСУБД “SQL Delta” [Электронный ресурс]. Режим доступа: https://www.sqldelta.com (Дата обращения: 04.12.2018).

10. Официальная документация к инструменту сравнения таблиц РСУБД “SQL Manager” [Электронный ресурс]. Режим доступа: https://www.sqlmanager.net (Дата обращения: 03.12.2018).

11. Официальная документация РСУБД “Oracle Database” [Электронный ресурс]. Режим доступа: https://www.oracle.com/ru/database/ (Дата обращения: 10.02.2019).

12. Сайт DB-engines [Электронный ресурс]. Режим доступа: https://db-engines.com/en/ranking_trend (дата обращения 27.04.2019).

13. Сайт Seagate [Электронный ресурс]. Режим доступа: https://www.seagate.com/files/www-content/our-story/trends/files/idc-seagate-dataage-whitepaper.pdf (дата обращения: 25.04.2019).

14. Сайт Searchqlserver [Электронный ресурс]. Режим доступа: https://searchsqlserver.techtarget.com/definition/database (дата обращения: 23.05.2019)

15. Сайт W3techs. Trends in the usage of server-side languages for websites [Электронный ресурс]. Режим доступа: https://w3techs.com/ technologies/ history_overview/programming_language/m s/y (дата обращения 15.04.2019).

16. *Тишин А.О.* Разработка базы данных завершенных проектов // Евразийский научный журнал. 2017. № 5. С. 456–457.

CREATING A COMPARISON METHOD FOR RELATIONAL TABLES

A. S. Yakupov[1], D. A. Klinov[2]

[1]LLC “Piano”; [2]LLC “deLion”

[1]asyakupov@kpfu.ru, [2]daniil.klinov@delion.ru

Abstract

The article is devoted to creating a quick method of comparing a huge amount of data tables in relational database management systems. Creating an effective method for comparing relational systems is really relevant today. The study of existing solutions was conducted. The algorithm in this article was created using the probabilistic data structure «Countable Bloom filter» and the Monte Carlo Method. The proposed solution is unique in its direction, as it uses the least amount of temporary resources. A probabilistic model of the created algorithm is constructed, this algorithm can be used for parallelization.

Keywords: *multiset, comparison of relational tables, heterogeneous system, Countable Bloom filter, Monte Carlo method, replication, Oracle, PostgreSQL, Probabilistic data structure*

REFERENCES

1. *Birialtsev E.* Intelligent search in Big Data // Approach to Data Integration. 2017. V. 46. No 19. P. 7–14.

2. *Chen G., Guo D., Luo L., Ren B.* Optimization of multicast source routing based on bloom filter // IEEE Communication Letters. 2018. No 4. P. 700–703.

3. *Kareev I.* Lower bounds for expected sample size of sequential procedures for the multinomial selection problems // Communications in Statistics. 2017. V. 913. No 1. P. 1–29.

4. *Wu K. , Tan H., Liu Y., Zhang J., Zhang Q., Ni L.* Side channel: Bits over interference // IEEE Transactions on Mobile Computing. 2017. No 8. P. 1317–1330.

5. *Afanasev G.I., Markov A.D.* Baza Dannyh NoSql i ih sravnenie s tradicionnymi bazami dannyh // Teoriya innovazii. 2017. No 5-2. S. 4–10.

6. Official documentation for the table comparison tool for RDMS “Devart” [Internet resource]. Access mode: https://www.devart.com (Date of the application: 19.11.2018).

7. Official documentation for the table comparison tool for RDMS “Idera” [Internet resource]. Access mode: https://www.idera.com (Date of the application: 17.01.2019).

8. Official documentation for the table comparison tool for RDMS “Red Gate” [Internet resource]. Access mode: https://www.red-gate.com (Date of the application: 11.11.2018).

9. Official documentation for the table comparison tool for RDMS “SQL Delta” [Internet resource]. Access mode: https://www.sqldelta.com (Date of the application: 04.12.2018).

10. Official documentation for the table comparison tool for RDMS “SQL Manager” [Internet resource]. Access mode: https://www.sqlmanager.net (Date of the application: 03.12.2018).

11. Official documentation for RDMS “Oracle Database” [Электронный ресурс]. Access mode: https://www.oracle.com/ru/database/ (Date of the application: 10.02.2019).

12. Site DB-engines [Internet resource]. Access mode: https://db-engines. com/en/ranking_trend (Date of the application: 27.04.2019).

13. Site Seagate [Internet resource]. Access mode: https://www.seagate. com/ (Date of the application: 25.04.2019).

14. Site Searchqlserver [Internet resource]. Access mode: https://searchsqlserver. techtarget.com/definition/database (Date of the application: 23.05.2019)

15. Site W3techs. Trends in the usage of server-side languages for websites [Internet resource]. Access mode: https://w3techs.com/technologies/ (Date of the application: 15.04.2019).

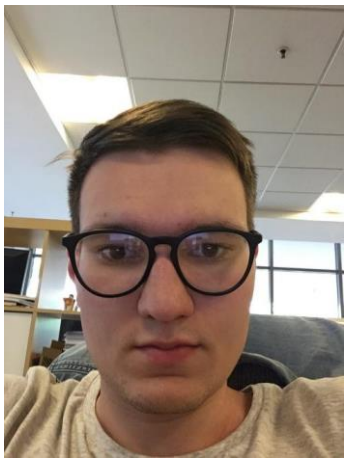
16. *Tishin A.O.* Razrabotka bazy dannyh zavershennyh proektov // Evraziiskii nauchnyi zhurnal. 2017. No 5. S. 456–457.

СВЕДЕНИЯ ОБ АВТОРАХ

ЯКУПОВ Азат Шавкатович – ассистент кафедры «Программная инженерия» Высшей школы информационных технологий и интеллектуальных систем, специалист в области баз данных.

Azat Shavkatovich YAKUPOV – Assistant at the department of “Software engineering” HS ITIS, senior of database developing.

asyakupov@kpfu.ru



КЛИНОВ Даниил Андреевич – бакалавр Высшей школы информационных технологий и интеллектуальных систем по направлению «Прикладная информатика».

Daniil Andreevich KLINOV – the bachelor of HS ITIS in the direction “Applied informatics”, junior of database developing.

daniil.klinov@delion.ru

Материал поступил в редакцию 29 июня 2019 года