МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
**«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИМЕНИ Н. Г. ЧЕРНЫШЕВСКОГО»**
Кафедра математической кибернетики и компьютерных наук

РАЗРАБОТКА ВЕБ-ПРИЛОЖЕНИЯ ДЛЯ РАБОТЫ С БАЗОЙ ДАННЫХ И ДЛЯ ХРОНОЛОГИЗАЦИИ ИСТОРИЧЕСКИХ СОБЫТИЙ

АВТОРЕФЕРАТ БАКАЛАВРСКОЙ РАБОТЫ

студента 4 курса 451 группы
направления 09.03.04 — Программная инженерия
факультета КНиИТ
Левшунова Михаила Александровича

Научный руководитель
доцент, к. ф.-м. н. _____ А. С. Иванова

Заведующий кафедрой
к. ф.-м. н. _____ С. В. Миронов

Саратов 2019

ВВЕДЕНИЕ

В последние годы информационные технологии все чаще используются во всех отраслях знания. Уже тяжело представить нашу жизнь без смартфонов, распознавания лиц или системы умного дома. Информационные технологии играют очень важную роль в научных исследованиях и при разработке проектов любого рода. Исследования в области истории не стали исключением.

Одной из задач исторической науки, требующей решения с использованием информационных технологий, является задача восстановления хронологической последовательности избрания магистратов в древнегреческих полисах [1, 2]. Один из подходов к решению этой задачи был рассмотрен в работе [3].

В древнем мире магистраты (астиномы) выполняли роль чиновников, ответственных за экономическую и политическую жизнь общества. Функции астиномов описаны в работах многих историков [4–6]. В частности, магистраты принимали участие в контроле керамического производства.

В период античности на берегах Черного моря было разбросано более пятисот греческих полисов, ведущих активную торговлю друг с другом. В этих полисах для перевозки на кораблях вина и оливкового масла использовались керамические амфоры. Чаще всего эта тара была одноразовой и, благодаря этому, ее производили в огромных количествах.

В ходе раскопок на черноморском побережье было обнаружено значительное количество остатков таких амфор [7–9]. Кроме того, многие амфоры маркировались одним или двумя клеймами, в которых фигурируют имена магистрата, занимавшего описанную выше должность во время создания этой амфоры, и фабриканта, владевшего мастерской, в которой эта тара была произведена. Подробнее об этом можно прочитать в книгах и статьях Монахова С. Ю., Федосеева Н. Ф., Каца В. И. [7–9].

Рассмотрим задачу восстановления хронологической последовательности имен магистратов в порядке их избрания. Историки считают, что каждый магистрат исполнял обязанности только один срок и не был переизбран. При решении задачи будем предполагать, что все найденные изделия предоставляют полную информацию для выбранного промежутка времени, т. е. если некоторый фабрикант изготовлял амфоры в период выполнения обязанностей некоторым магистратом, то хотя бы одна такая амфора обнаружена археолога-

ми и загружена в базу данных.

Целью дипломной работы является создание веб-приложения для работы с базой данных описанных выше амфор, магистратов и фабрикантов. Приложение должно поддерживать возможность сохранения, просмотра и обработки данных о найденных археологами амфорах и клеймах на них в виде ссылочной структуры с возможностью быстрого получения информации о магистратах и фабрикантах, указанных на этих клеймах. Также должна быть возможность группировать данные по полисам, исторической дате или просто для удобства обработки. Кроме того, приложение должно решать поставленную выше задачу хронологизации и поддерживать работу большого количества пользователей и разделять их по ролям в системе с выдачей соответствующих прав.

1 Задача хронологизации данных

1.1 Поиск решения

Для представления информации о найденных археологами амфорах будем использовать матрицу $M \times N$, где M есть количество идентификаторов фабрикантов, а N есть количество идентификаторов магистратов. Столбцы этой матрицы характеризуют магистратов, а строки – фабрикантов. Если была найдена амфора, изготовленная фабрикантом с номером i, когда должность занимал магистрат с номером j, то на пересечении строки i и столбца j будет стоять 1, в противном случае – 0.

Необходимо переставить столбцы матрицы в порядке, который соответствует предполагаемому порядку избрания астиномов.

Для решения поставленной задачи будем отталкиваться от предположения, что каждый фабрикант работал в некоторый непрерывный промежуток времени, который мог охватить время исполнения обязанностей несколькими магистратами. Таким образом, в матрице необходимо добиться того, что в каждой строке все единицы образуют один непрерывный отрезок.

Для решения задачи мы будем использовать следующую эвристику: два магистрата занимали должность в соседние промежутки времени тогда, когда максимизировано количество фабрикантов, которые работали при обоих этих магистратах.

Обозначим через $f(u, v),\ u, v \in 1, \ldots, N$ количество фабрикантов, которые работали как при астиноме u, так и при астиноме v. Для общности будем считать, что $\forall u \in 1, \ldots, N\ f(u, u) = 0$.

Тогда оптимальная последовательность имен магистратов должна максимизировать сумму значений функции $f(u, v)$ по всем парам соседних магистратов. Другими словами, среди всех возможных хронологических последовательностей избрания $u_1, u_2, \ldots, u_N$, оптимальной будет та, которая максимизирует следующую целевую функцию:

$$\sum_{i=1}^{N-1} f(u_i, u_{i+1}) \to \max. \tag{1}$$

Заметим, что данные можно представить в виде полного взвешенного графа [10] из N вершин, где вершинами будут метки магистратов, а вес ребра между вершинами u и v – значение функции $f(u, v)$. Петли нулевого веса в

этот граф добавлять не будем, так как при $\forall u \in 1, \ldots, N \ f(u, u) = 0$, они никак не влияют на решение.

В таком графе оптимальным будет путь, который проходит через каждую вершину графа ровно один раз и при этом сумма весов ребер на этом пути максимальна.

В теории графов путь, который проходит по всем вершинам графа ровно один раз, называется гамильтоновым путем [11]. А задача максимизации суммы весов ребер на таком пути является частным случаем задачи коммивояжера [12].

Тот факт, что такое сведение задачи корректно и позволяет построить необходимую последовательность, доказывается в работе [13], написанной в соавторстве с Мироновым С.В., Файзлиевым А.Р. и Сидоровым С.П.

1.2 Задача коммивояжера

Задача коммивояжера [14] представляет собой задачу отыскания в полном взвешенном графе кратчайшего пути, который проходит через все вершины ровно один раз.

В теории алгоритмов задача коммивояжера относится к классу NP-трудных задач, а соответственно ее точное решение не может быть получено за полиномиальное время. Однако, для решения задачи коммивояжера существует множество различных алгоритмов.

В приложении реализуются два алгоритма. Один точный для более качественной хронологизации небольших объемов данных, и один быстрый для работы со значительными объемами.

1.2.1 Алгоритм Беллмана–Хелда–Карпа

В качестве точного алгоритма был выбран алгоритм Беллмана–Хелда–Карпа как наиболее эффективный из этого класса. Кроме того, именно этот алгоритм используется в работе [13], описанной выше.

Алгоритм Беллмана–Хелда–Карпа [15] строит путь последовательно, выбирая лучший переход. Он позволяет сократить количество перебираемых вариантов за счет того, что при выборе очередного перехода не важно, как именно путь шел до него, и как пойдет после. Необходимо знать лишь длину этих путей и то, какие вершины содержатся в них.

Этот алгоритм использует метод динамического программирования и его асимптотика равна $O(n \cdot 2^n)$. Однако, в отличии от алгоритмов, описанных выше, он находит решение для заданной начальной вершины. Поэтому, если стартовая вершина неизвестна, ее необходимо перебрать, и это увеличивает временную сложность до $O(n^2 \cdot 2^n)$.

При использовании данного алгоритма необходимо учитывать, что его оригинальная версия рассчитана на задачу коммивояжера с заранее известной стартовой вершиной и целью минимизации стоимости пути. Однако эти моменты легко дорабатываются.

На каждом шаге алгоритма необходимо решить набор из $N \cdot 2^N$ подзадач, на решение каждой из которых необходимо время, пропорциональное $O(N)$. Шагов всего так же N, таким образом решение работает за асимптотику $O(N^3 \cdot 2^N)$.

Кроме того, решение требует $O(N^2)$ дополнительной памяти на сохранение графа и еще $O(N \cdot 2^N)$ на сохранение массива состояний. Это приводит к тому, что уже при $N = 22$ на решение поставленной задачи требуется чуть больше одного гигабайта оперативной памяти. И при увеличении N это число растет на порядок.

Полученное ограничение на $N = 22$ является проблемой. Более того, для поддержания стабильной работы всего приложения, это ограничение необходимо опустить еще сильнее, для этого было выбрано значение $N = 15$. Однако, несмотря на ограничения, алгоритм позволяет получить максимально возможную точность решения поставленной задачи для небольших блоков данных, поэтому реализуется в приложении.

1.2.2 Муравьиный алгоритм

Для хронологизации больших объемов данных требуется более быстрый и эффективный по памяти алгоритм. Лучшим алгоритмом для получения достаточно неплохого решения за тот короткий срок, что готовы ждать пользователи приложения, является муравьиный алгоритм, поэтому для реализации был выбран именно он.

Муравьиный алгоритм [16] использует идею роевого интеллекта колонии муравьев. Несмотря на крайне примитивное устройство каждого отдельно взятого муравья, их вид успешен, существует на планете более 100 миллионов лет, строит огромные жилища, обеспечивает их всем необходимым и даже

ведет настоящие войны. Таких успехов муравьи достигают благодаря своей социальности, они живут только в коллективах – колониях. Все муравьи колонии формируют так называемый роевой интеллект. Особи, составляющие колонию, не должны быть умными: они должны лишь взаимодействовать по определенным крайне простым правилам, и тогда колония целиком будет эффективна.

Алгоритм моделирует поведение колонии муравьев в процессе решения поставленной задачи. Так же как и генетический алгоритм, он позволяет получать очень хорошее решение. Однако, его важным отличием является тот факт, что он может очень быстро достичь решения, близкого к идеальному, а затем очень долго идти к этому идеалу. Так как в процессе работы приложения пользователь не захочет ждать, пока алгоритм найдет лучшее решение, муравьиный алгоритм идеально подходит в качестве способа быстро получить решение, пожертвовав лишь небольшой долей точности.

В качестве основы для написания алгоритма была взята статья [17] с анализом различных вариаций муравьиного алгоритма и подбором лучших параметров для решения задачи коммивояжера. Единственным важным отличием является то, что в статье решается задача на минимум, а для хронологизации необходимо решение задачи на максимум, но и эту проблему можно преодолеть.

На каждую итерацию алгоритма необходимо время, пропорциональное $O(N^2m)$. Но так как в каждую вершину в начале каждой итерации бросается ровно один муравей, их количество совпадает с количеством вершин и асимптотику можно переписать как $O(N^3)$. Кроме того, алгоритму необходимо $O(N^2)$ памяти на сохранение графа и массивов видимостей и феромонов.

Полученная асимптотика значительно лучше, чем в предыдущем алгоритме и позволяет проводить хотя бы 10 итераций алгоритма за секунду при $N \leqslant 200$. Добиться лучших результатов на данный момент теоретически не возможно, поэтому остановимся на этом.

2 Реализация приложения AmphoraWebApp

2.1 База данных

С учетом требования работы приложения с большим количеством пользователей, в базе данных была создана соответствующая сущность. Пользователь характеризуется его логином (Username), ФИО (Name), паролем и одной из доступных ролей.

Основной задачей приложения является работа с массивами амфор, магистратов и фабрикантов, поэтому в базе данных должны присутствовать сущности для каждого из них. В базе хранится уникальный идентификатор объекта, его название и описание. Кроме того, для каждого из объектов хранится ссылка на пользователя, добавившего этот объект в базу и пометка о том, доступен ли этот объект кому-либо, кроме пользователя, добавившего его. Для амфор также хранятся ссылки на магистрата и фабриканта, чьи клейма стояли на данной амфоре.

Для группировки данных используется структура под названием Эпоха. Эпоха может характеризовать как какой-то временной промежуток, так и территориальный признак, а также просто использоваться для удобства разделения данных. Аналогично другим объектам, у эпохи есть название, описание, ссылка на создавшего пользователя и пометка о видимости этой эпохи.

Каждый объект может принадлежать к любому количеству различных эпох. Для сохранения этих связей используются три отдельные таблицы, каждый элемент которых характеризует связывание какого-либо объекта с определенной эпохой.

2.2 Защита приложения и аутентификация

Для защиты приложения используется фреймворк Spring Security. Он берет на себя большую часть забот об организации безопасной работы.

Кроме того, в приложении используется csrf-защита. Смысл этой защиты заключается в том, что если злоумышленник попытается отправить запрос в обход пользовательской формы, то без csrf-токена сервер откажется принять этот запрос.

Каждый раз, когда к приложению осуществляется доступ с нового браузера или устройства, а также при каждой успешной авторизации, формируется уникальный 32-значный код, идентифицирующий сессию данного пользова-

теля. Этот код сохраняется в куках (cookie) браузера клиента под названием JSESSIONID и автоматически передается при каждом запросе к серверу. Если кода в запросе нет, то сервер создает новую сессию и отправляет новый код браузеру пользователя. Если же кука сервером получена, он находит по ней информацию по текущему пользователю.

Это не единственная кука, сохраняемая приложением в процессе использования. Для получения доступа к большинству функций, необходимо выбрать одну из эпох, доступных в приложении. После этого, название выбранной эпохи будет отображаться на каждой странице, а все url-адреса на списки объектов эпохи будут формироваться с учетом нее. Однако, в отличии от JSESSIONID, она не формируется и не подставляется автоматически.

2.3 Функциональность

На каждой странице приложения присутствует навигационная панель со ссылками на все основные страницы:

- AmphoraWebApp – логотип приложения, ведет на главную страницу.
- Главная – стартовая страница приложения с приветствие и кратким руководством пользователя.
- Данные – на этой странице присутствуют 3 независимые таблицы, включающие в себя всех магистратов, фабрикантов и все амфоры выбранной эпохи.
- Таблица – страница с большой сводной таблицей по всем объектам эпохи.
- Добавить – ведет на динамическую страницу, на которой можно добавить любой новый объект или эпоху.
- Пользователи – таблица со списком всех зарегистрированных пользователей приложения.
- Текущая эпоха: Название эпохи – отображает выбранную на данный момент эпоху, при клике открывает страницу с информацией об этой эпохе.
- Сменить эпоху – ведет на страницу со списком всех доступных пользователю эпох.
- Логин пользователя – ведет на страницу с информацией по пользователю.
- Выйти – кнопка выхода из учетной записи.

До авторизации большей части этих функций недоступно: остается толь-

ко логотип с главной страницей и предложение авторизоваться с кнопкой «Войти».

2.3.1 Данные

На странице «Данные» находятся 3 таблицы, в каждой из которых можно увидеть всех магистратов, амфоры и фабрикантов этой эпохи соответственно. Объекты в каждой таблице отсортированы лексикографически по названию или имени соответственно. Каждая таблица разделена на страницы по 10 объектов на каждой странице. Страницы можно переключать с помощью специальной утилиты внизу таблицы, при этом если страниц слишком много, часть из них заменяется многоточием. Текущие номера страниц постоянно передаются между клиентом и сервером с использованием параметров http-запросов и класса Model.

Для разделения объектов по страницам используется класс Pageable, предоставляемой фреймворком Spring Data. Он позволяет при каждом запросе получать из базы только ограниченный набор данных, учитывая заданную сортировку, количество элементов на странице и номер страницы. Все поля каждой таблицы кликабельны.

2.3.2 Таблица

На странице «Таблица» находится большая сводная таблица по всем магистратам, фабрикантам и амфорам эпохи. Столбцы характеризуют магистратов, строки – фабрикантов, а число на пересечении указывает на количество амфор, одновременно содержащих клейма соответственных магистрата и фабриканта.

Строки и столбцы пронумерованы, а при клике на номер появляется плашка с описанием соответствующего магистрата или фабриканта.

Таблица может увеличиваться до бесконечности, поэтому может не помещаться на экране, если в эпохе присутствует значительное количество магистратов и фабрикантов.

Все магистраты в таблице хронологически упорядочены. При переходе на эту страницу приложение проверяет, были ли изменения в эпохе после последнего запуска алгоритма. Если изменения были – быстро запускает алгоритм, сохраняет полученный результат и выводит таблицу исходя из него. Если изменений не было и упорядочивание сохранено в базе – строит таблицу

на его основе.

Кроме того, фабриканты также упорядочиваются таким образом, чтобы амфоры в таблице формировали плавную лесенку.

2.3.3 Пользователи

На странице «Пользователи» можно увидеть список всех пользователей, зарегистрированных в приложении. Аналогично всем остальным таблицам, длинные тексты обрезаются и доступны по клику, а номер является ссылкой на отдельную страницу с этим пользователем.

Уникальной же особенностью данная таблица обладает, если ее просматривает один из администраторов. В этом случае, для пользователей с меньшими правами, чем у этого администратора, вместо отображения роли появляются кнопки вида radio button для изменения роли данного пользователя. После выбора подходящей роли, администратору необходимо нажать на кнопку «Сохранить», находящуюся рядом со списком доступных ролей.

2.3.4 Список эпох

При клике на кнопку «Сменить эпоху» на навигационной панели, можно попасть на страницу с таблицей, в которой расположен список всех эпох, доступных пользователю. Для администраторов здесь будут вообще все созданные эпохи, а обычные пользователи не будут видеть приватные эпохи других пользователей. В дополнение к стандартным функциям таблиц, в этой расположено 2 дополнительные кнопки: «Переключиться», при нажатии на которую текущая выбранная эпоха сменится на ту, напротив которой расположена нажатая кнопка, а вверху появится предупреждение об успешной смене эпохи; и «Удалить», доступная администраторам для всех эпох, а обычным пользователем только для приватных, созданных ими самими.

2.3.5 Эпоха

При нажатии на кнопку «Текущая эпоха: Название эпохи» на навигационной панели, либо при переходе к конкретной эпохе из их списка, откроется страница с подробным описанием выбранной эпохи. На странице можно просматривать и, если достаточно прав, изменять название эпохи и ее описание. Кроме того здесь отображается создатель эпохи, является ли она приватной или открытой для всех, и, если она приватная, то создатель эпохи может здесь

сделать ее доступной глобально. При нажатии на соответствующую кнопку появится предупреждение, аналогичное тому, что появляется при попытке удалить какой-либо объект.

Кроме того, на странице эпохи можно перейти к списку объектов и сводной таблице, эти кнопки являются синонимами тех, что расположены на навигационной панели. Также в самом низу доступна кнопка «Переключиться», аналогичная той, что расположена на странице со списком эпох.

2.3.6 Магистрат и фабрикант

При кликах на номер магистрата или фабриканта на страницах с данными, либо при нажатии кнопки «Показать отдельно» на соответствующих всплывающих плашках, открывается страница с описанием выбранного магистрата или фабриканта. Данные страницы практически полностью идентичны – отличается только название объекта в ее левой верхнем углу.

Вся страница разделена на 3 части: в левой части находятся основные элементы выбранного объекта, такие как имя, описание, ссылка на добавившего пользователя и кнопки для редактирования этой информации (если у пользователя достаточно прав для этого). В центральной части страницы расположен список всех амфор, на которых были обнаружены клейма выбранного фабриканта или магистрата. Таблица с амфорами аналогична той, что расположена на основной странице с данными. Наконец в правой части расположена таблица со списком всех эпох, в которых присутствует данный объект. В каждой строке содержится информация о соответствующей эпохе, а также кнопка для исключения объекта из этой эпохи.

2.3.7 Амфора

Страница с описание конкретной амфоры во многом похожа на страницы магистрата и фабриканта. Доступ к ней можно получить также со страницы с данными, на ней также присутствует описание основных элементов в левой части страницы, однако здесь еще добавлены ссылки на магистрата и фабриканта, чьи клейма были найдены на этой амфоре. В правой части страницы также можно увидеть список эпох. Основным отличием страницы с амфорой является тот факт, что она разделена всего на 2 части, они шире, а никакой дополнительной таблицы по середине нет.

2.3.8 Пользователь

Страница пользователя доступна при нажатии на кнопку «Открыть отдельно» на плашках, открывающихся при клике на ссылку на пользователя на различных страницах, а также при нажатии на имя пользователя на навигационной панели.

Страница пользователя по структуре напоминает страницу с описанием амфоры. Однако в правой ее части находится не список эпох, к которым относится объект, а список эпох, добавленных пользователем. Кроме того, на этой странице доступно удаление добавленных эпох.

В левой части страницы для всех пользователей доступен просмотр логина, ФИО и роли пользователя, а также кнопка с ссылкой на особую страницу с данными, на которой можно увидеть все амфоры, всех магистратов и всех фабрикантов, добавленных пользователем. Левая часть страницы значительно изменяется в зависимости от того, какой пользователь зашел на нее. Для владельца аккаунта кроме описанных выше полей появляются 3 поля для пароля: для установки нового с подтверждением, а также для ввода старого, которым на этой странице нужно подтверждать все изменения. Для администратора видоизменяется строка с ролью – на ее месте появляются кнопки с возможностью установить роль данному пользователю.

ЗАКЛЮЧЕНИЕ

В результате работы было создано приложение, позволяющее работать с базой данных древнегреческих амфор, магистратов и фабрикантов. Оно позволяет просматривать совокупную базу двумя удобными способами, а также изучать, редактировать, добавлять и удалять объекты по одиночке. Приложение поддерживает работу (в т. ч. одновременную) нескольких пользователей с возможность разделения ролей.

Также разработанное приложение позволяет решать проблему исторической науки, связанную с восстановлением хронологической последовательность избрания магистратов, используя сведение этой проблемы к задаче комивояжера.

В перспективе возможно размещение приложения на постоянном сервере. Это позволит собрать общую базу данных от пользователей со всего мира.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1 *Федосеев, Н.Ф.* О хронологии синопских керамических клейм / Н.Ф. Федосеев // *Античный мир и археология*. — 2015. — no. 17. — Pp. 352–364.

2 *Fedoseev, N.F.* Classification des timbres astynomiques de sinope / N.F. Fedoseev // *Production et Commerce des Amphores anciennes en Mer Noire*. — 1999. — no. 17. — Pp. 27–48.

3 *Бобылев, А.Б.* Экспертный пакет АСТИНОМ / А.Б Бобылев, Н.Ф. Федосеев // *Методы и системы технической диагностики*. — 1989. — Vol. 1, no. 12. — Pp. 112–115.

4 *Badoud, N.* The contribution of inscriptions to the chronology of rhodian amphora eponyms / N. Badoud // *Pottery, Peoples and Places: The Late Hellenistic Period, c. 200–50 BC Between the Mediterranean and the Black Sea*. — 2014. — Pp. 17–28.

5 *Badoud, N.* Deciphering greek amphora stamps / N. Badoud // *CHS Research Bulletin 5*. — 2017. — no. 2.

6 *Lawall, M.L.* Amphoras and hellenistic economies: Addressing the (over) emphasis on stamped amphora handles / M.L. Lawall // *In Making, moving, and managing: The new world of ancient economies, 323–31 BC*. — 2005. — Pp. 188–232.

7 *Федосеев, Н.Ф.* Из истории Синопы. Керамический аспект / Н.Ф. Федосеев // *Таврические студии. Исторические науки*. — 2014. — no. 6. — Pp. 90–97.

8 *Кац, В.И.* Керамические клейма Херсонеса Таврического / В.И Кац. — Саратов: Издательство Саратовского университета, 1994.

9 *Монахов, С.Ю.* Греческие амфоры в причерноморье. Комплексы керамической тары VII–II веков до н.э. / С.Ю. Монахов. — Саратов: Издательство Саратовского университета, 1999.

10 *Харари, Ф.* Теория графов / Ф. Харари, В.П. Козырев, Г.П. Гаврилов. — М.: Мир, 1973.

11 *Седжвик, Р.* Фундаментальные алгоритмы на C++: Алгоритмы на графах. Часть 5 / Р. Седжвик. — М.: Диа Софт, 2002.

12 *Левитин, А.В.* Алгоритмы: введение в разработку и анализ. / А.В. Левитин. — М.: Издательский дом Вильямс, 2006.

13 *Левшунов, М.А.* Restoring the succession of magistrates in ancient greek poleis: How to reduce it to travelling salesman problem using heuristic approach / М.А. Левшунов, С.В. Миронов, А.Р. Файзлиев, С.П. Сидоров // *Social Informatics*. — 2018. — Vol. 1. — Pp. 336–347.

14 *Борознов, В.О.* Исследование решения задачи коммивояжера / В.О. Борознов // *Вестник Астраханского государственного технического университета. Серия: управление, вычислительная техника и информатика*. — 2009. — Pp. 147–151.

15 *Р., Беллман*. Применение динамического программирования к задаче о коммивояжере / Беллман Р. // *Кибернетический сборник*. — 1964. — Vol. 9. — Pp. 219–228.

16 *Dorigo, M.* Ottimizzazione, apprendimento automatico, ed algoritmi basati su metafora naturale: Ph.D. thesis / Politecnico di Milano. — 1992.

17 *Dorigo, M.* The ant system: Optimization by a colony of cooperating agents / M. Dorigo, V. Maniezzo, A. Colorni // *IEEE Transactions on Systems, Man, and Cybernetics-Part B*. — 1996. — Vol. 26, no. 1. — Pp. 29–41.