**МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ**

**Белорусский государственный Университет**

**ФАКУЛЬТЕТ ПРИКЛАДНОЙ МАТЕМАТИКИ И ИНФОРМАТИКИ**

**Кафедра многопроцессорных систем и сетей**

МИРЕЙКО

Наталья Викторовна

**РЕАЛИЗАЦИЯ СИСТЕМЫ ПРОГНОЗИРОВАНИЯ**

**ПРОДАЖ ДЛЯ КРУПНОГО БИЗНЕСА**

Дипломная работа

|  |  |  |
| --- | --- | --- |
|  |  | Научный руководитель:  ассистент кафедры МСС  А.С. Гусейнова |
|  |  |  |

Допущена к защите

«\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_2017 г.

Зав. кафедрой многопроцессорных систем и сетей

доцент, кандидат физико-математических наук С.В. Марков

Минск, 2017

**РЕФЕРАТ**

Дипломная работа, 34 с., 15 рис., 1 табл., 6 формул, 8 источников.

*Ключевые слова*: ПРОДАЖИ, МУЛЬТИПЛАТФОРМЕННОСТЬ, ЗАДАЧА ПРОГНОЗИРОВАНИЯ, МОДЕЛИ И МЕТОДЫ ПРОГНОЗИРОВАНИЯ, МЕТОД ПРОГНОЗИРОВАНИЯ ХОЛЬТА-ВИНТЕРСА, ТРЕНД, СЕЗОННОСТЬ.

*Объект исследования* – концепции и методы мульти-платформенной разработки, модели и методы прогнозирования, применение метода Хольта-Винтерса в задачах прогнозирования, системы прогнозирования продаж для крупного бизнеса.

*Методы исследования* – обзор и изучение методов и моделей прогнозирования, обзор и изучение метода Хольта-Винтерса, получение теоретических и практических знаний в гибкой мульти-платформенной разработке.

*Цель работы* – разработка системы прогнозирования продаж для крупного бизнеса на основе математических моделей теории прогнозирования.

*Результатом* является реализованная система прогнозирования продаж для крупного бизнеса, изучение, освоение и реализация метода Хольта-Винтерса в задачах прогнозирования, изучение и освоение моделей и методов прогнозирования.

*Областью применения* являются системы автоматизированного анализа и прогнозирования продаж для крупного бизнеса.

**РЕФЕРАТ**

Дыпломная праца, 34 с., 15 мал., 1 табл., 7 формул, 8 крыніц.

*Ключавыя словы*: ПРОДАЖЫ, МУЛЬТЫПЛАТФОРМЕНАСЦЬ, ЗАДАЧА ПРАГНАЗАВАННЯ, МАДЭЛІ І МЕТАДЫ ПРАГНАЗАВАННЯ, МЕТАДЫ ПРАГНАЗАВАННЯ ХОЛЬТА-ВІНТЭРСА, ТРЭНД, СЕЗОННАСЦЬ.

*Аб'ект даследавання –* канцэпцыі і метады мульты-платформавай распрацоўкі, мадэлі і метады прагназавання, прымяненне метаду Хольта-Вінтэрса ў задачах прагназавання, сістэмы прагназавання продажаў для буйнога бізнэсу.

*Метады даследавання* - агляд і вывучэнне метадаў прагназавання, агляд і вывучэнне метаду Хольта-Вінтэрса, атрыманне тэарэтычных і практычных ведаў у гнуткай мульты-платформавай распрацоўцы.

*Мэта працы* - распрацоўка сістэмы прагназавання продажаў для буйнога бізнэсу на аснове матэматычных мадэляў тэорыі прагназавання.

*Вынікам* з'яўляецца рэалізаваная сістэма прагназавання продажаў для буйнога бізнэсу, вывучэнне, засваенне і рэалізацыя метаду Хольта-Вінтэрса ў задачах прагназавання, вывучэнне і засваенне мадэляў і метадаў прагназавання.

*Вобласцю ўжывання* з'яўляецца сістэмы аўтаматызаванага аналізу і прагназавання продажаў для буйнога бізнэсу.

**ABSTRACT**

Diploma work, 34 pages, 15 pictures, 1 tables, 7 formulas, 8 sources.

*Key words*: SALES, DIFFERENT PLATFORMS, PREDICTION ISSUE, METHODS AND MODELS OF PREDICTION, HOLT-WINTERS PREDICTION, TREND, SEASONALITY.

*Object of research* – concepts and methods of development involving different technologies, models and methods of prediction, the applying of Holt-Winters method for prediction issue, systems of sales prediction for big business.

*Research* *methods* – learning methods and models of prediction, learning the Holt-Winters method, getting the theoretical and practical skills in flexible development with different technologies.

*Purpose* – development of system for forecasting sales for big business based on mathematical models of prediction theory.

*The result* of diploma work is an implemented system for forecasting sales for big business, learning and implementation of Holt-Winters method for prediction issue, learning methods and models of prediction.

*The field of application* is automated analysis and forecasting systems for big business.

**ОГЛАВЛЕНИЕ**

[ВВЕДЕНИЕ 6](#_Toc452301570)

АРХИТЕКТУРА СИСТЕМЫ И [ОБЗОР ИСПОЛЬЗУЕМЫХ ТЕХНОЛОГИЙ 8](#_Toc452301571)

[1.1 Архитектура системы 8](#_Toc452301572)

[1.2 Язык программирования С# 9](#_Toc452301573)

[1.3 Платформа ASP.NET Core 12](#_Toc452301574)

[1.4 Entity Framework 13](#_Toc452301575)

[1.5 База данных SQLite 13](#_Toc452301575)

[1.6 AngularJS 13](#_Toc452301575)

[МОДЕЛИ И МЕТОДЫ ПРОГНОЗИРОВАНИЯ 16](#_Toc452301576)

[2.1 Основные понятия теории прогнозирования 16](#_Toc452301577)

[2.2 Основные модели прогнозирования 17](#_Toc452301578)

[2.3 Сравнение моделей прогнозирования 21](#_Toc452301579)

[2.4 Метод экспоненциального сглаживания Хольта–Винтерса 27](#_Toc452301584)

[2.5 Применение метода Хольта–Винтерса к поставленной задаче 29](#_Toc452301585)

[ПОСТАНОВКА ЗАДАЧИ И РЕАЛИЗАЦИЯ 30](#_Toc452301586)

[3.1 Модель данных и алгоритм системы 30](#_Toc452301587)

[3.2 Описание функционала системы 33](#_Toc452301588)

[3.3 Интерфейс системы 40](#_Toc452301589)

[3.4 Обзор аналогов 44](#_Toc452301590)

[ЗАКЛЮЧЕНИЕ 55](#_Toc452301591)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 57](#_Toc452301592)

## ВВЕДЕНИЕ

Прогнозирование – актуальная тема на сегодняшний день. Независимо от сферы бизнеса компании и должностных обязанностей ее работников, у каждого из них существует потребность в прогнозе – какой-то оценке будущего, на основе которой можно построить план. Эти планы применяются ко многим различным областям компании, которые имеют разные потребности и задачи. Например, маркетологи нуждаются в прогнозах, чтобы определить, какие новые продукты или услуги следует вводить или прекращать поставлять, какие входят на рынок, а какие выходят из него, и какие продукты следует продвигать. Специалисты по продажам, используют прогнозы для составления планов продаж, так как коммерческие квоты обычно основаны на оценках будущих продаж. Менеджеры в рамках цепочки поставок используют прогнозы для составления производственных, закупочных и логистических планов. Специалисты по финансам используют прогнозы для составления финансовых планов. И таких примеров, где нужен прогноз, очень много.

Трудно количественно оценить преимущества прогнозирования, однако в отчетах по продажам есть определенные критерии, по которым можно количественно оценить преимущества прогнозирования. Одним из способов измерения преимуществ прогнозирования является определение того, какой доход был бы потерян, если бы прогноз был неточным. Другой способ измерить их - это то, сколько было бы получено или сохранено дохода с улучшенными прогнозами.

Важно отметить, что ошибки прогноза могут возникать из-за недооценки или чрезмерного прогнозирования. При недооценке в прогнозе потери компании возникают из-за потери потенциальных продаж регулярных и сопутствующих продуктов, на которые возрос спрос в определенный момент, и компании пришлось увеличить расходы на производство и доставку продуктов, чтобы скомпенсировать дефицит. В случае чрезмерного прогнозирования убытки компании возникают из-за скидок и акций, которые должны предлагаться для избавления от излишних запасов. Кроме того, устаревание, перегрузка, складские расходы и затраты на продвижение избыточных запасов также могут способствовать потерям дохода компании при чрезмерном прогнозировании, помимо расходов на возможную транспортировку продуктов из одного распределительного центра в другой для устранения избыточных запасов.

По данным источника [1], который опросил десять компаний (три в области компьютерных технологий и семь в области потребительских товаров), имеющих объем продаж от 100 миллионов до 3 миллиардов долларов США, улучшенные прогнозы с высокой точностью прогнозирования уменьшают потери и увеличивают сбережения компании. Для количественной оценки сбережений (или потерь) от недооценки прогнозирования опрошенные компании попросили предоставить информацию о своих наценках, процентах продаж, потерянных из регулярных и сопутствующих товаров в результате недооценки, и процентном увеличении затрат на производство и доставку для удовлетворения спроса на эти продукты. Для количественной оценке сбережений (или потерь) от чрезмерного прогнозирования, опрошенные компании попросили предоставить информацию о своих наценках, проценте запасов, которые будут продаваться со скидкой, и предлагаемых скидках, информацию о том, должны ли некоторые продукты транспортироваться из одного распределительного центра в другой, чтобы избавиться от них и затраты на перевозку, информацию о возможности устаревания определенного количества продуктов, стоимость утилизации устаревших продуктов, а также стоимость хранения избыточных запасов. В исследовании для каждой компании рассчитали экономию или убытки, обусловленные только 1-процентным улучшением точности прогнозирования. В результате было обнаружено, что компьютерные / технологические компании могут сэкономить в среднем 0,97 миллионов долларов США в год за счет уменьшения ошибки недооценки на 1 процентный пункт и сэкономить 1,58 миллионов долларов США в год за счет уменьшения такой же ошибки при чрезмерном прогнозировании. Для компаний потребительских товаров выгоды от 1-процентного улучшения точности прогнозирования были еще выше. Они могли бы экономить в среднем 3,52 миллионов долларов США в год в случае недооценки и 1,43 миллионов долларов США в случае чрезмерного прогнозирования. Разумеется, на ранних этапах разработки алгоритма системы прогнозирования компании могут уменьшить ошибку гораздо более, чем на 1 процентный пункт. Таким образом, экономия от функции прогнозирования и ее поддержки будет намного больше, чем показано в Приложении 1.

Таким образом, прогнозирование не просто необходимо. Это жизненно важно.

Однако если заранее знать о том, что вероятно возникновение задержки в тот или иной период, то можно увеличить работу двигателей и избежать задержки прибытия грузов.

Существующее программное обеспечение, которое можно найти в открытом доступе, обладает широким функционалом по управлению и распределению грузов, однако не обладает возможностью прогнозирования задержек на пути следования транспорта. Кроме того большинство подобного программного обеспечения оперирует только с наземным транспортом, и, в общем случае, решает задачу коммивояжера на определенной области карты, т.е. транспорт выбирает маршрут в зависимости от того, куда необходимо доставить грузы. В случае же морского транспорта маршрут планируется для судна заранее, а грузы привязываются к кораблю. Такая система возникла из-за множества факторов: объем перевозок, различные таможенные ограничения на корабли и грузы, экономия топлива, невозможность дозаправок и т.д. Именно поэтому у кораблей есть жесткий маршрут и расписание. Следовательно, при возникновении задержки могут возникнуть проблемные области, в которых перевозить грузы не выгодно. Таким образом, возникает проблема, как заранее определить возникновение задержек и как найти оптимальное распределение грузов по кораблям, при учете спрогнозированных задержек. Для прогноза задержек можно применять различные методы предсказания. Одним из них, конечно, является применение метода экспертных оценок, другим – использование численных методов решения задачи предсказания.

Использование анализа сингулярного спектра является хорошим и, на сегодняшний день, одним из самых популярных способов для решения задачи предсказания. Анализ сингулярного спектра (Singular spectrum analysis) – метод анализа временных рядов, основанный на преобразовании одномерного временного ряда в многомерный ряд с последующим применением к полученному многомерному временному ряду метода главных компонент. С помощью применения метода анализа сингулярного спектра, становится возможным прогнозирование задержек на основе предыдущих показателей. С учетом спрогнозированной задержки возникает вопрос, как распределить грузы по кораблям, чтобы уменьшить стоимость перевозок. Для решения этой задачи можно применить алгоритмы поиска оптимального решения.

На сегодняшний день генетические алгоритмы применяются во многих и различных сферах для поиска оптимальных решений. Интерес к использованию генетических алгоритмов растет вместе с ростом производительности вычислительной техники. Генетические алгоритмы могут применяться для решения тяжелых задач с большими объемами неизвестных коэффициентов и большими объемами вычислений. Генетический алгоритм (англ. genetic algorithm) — это эвристический алгоритм поиска, используемый для решения задач оптимизации и моделирования путём случайного подбора, комбинирования и вариации искомых параметров с использованием механизмов, аналогичных естественному отбору в природе.

Таким образом, в данной работе была поставлена задача – создать логистическую систему для морского транспорта, которая бы предоставляла возможности в просмотре местоположения кораблей и портов на карте. Так же предоставляла возможность, отображения того, какое количество грузов разгружается и загружается на корабль в каждом порту. Необходимо реализовать хранение и отображение информации о задержках кораблей. Так же необходима возможность предсказания задержек для морского транспорта. Поставлена задача в построении алгоритма поиска оптимального распределения грузов по кораблям при заданных портах, кораблях, маршрутах, определенном расписании кораблей и сведениях о грузах. Предсказание задержек необходимо реализовать с помощью алгоритма SSA. Поиск оптимального распределения грузов произвести с применением генетического алгоритма.

**ГЛАВА 1**

# ОБЗОР ИСПОЛЬЗУЕМЫЕХ ТЕХНОЛОГИЙ

Gen-technology Informational System (ГИС) – представляет собой мульти платформенную систему для создания надежных клиент-серверных приложений. С помощью данной системы разрабатываются как клиентская часть, так и серверная часть. Информационная система ГИС архитектурно состоит из двух частей – это само приложение и компоненты, с которыми данное приложение общается.

### 1.1 Разработка приложений

В данной информационной системе Клиентская часть, называемая Graphical User Interface (GUI), может представлять собой Web-страницу или же оконное приложение. Клиентская часть полностью реализуется на языке Java, разработка приложений на данной системе происходит в среде Eclipse, с помощью применения соответствующих плагинов. Для каждого GUI реализуется свой *контроллер*, который прослушивает java-компоненты данного приложения. Контроллер представляет собой Java класс, в котором описано множество методов для обработки стандартных Java событий java.awt.AWTEvent[1]. Методы, реализованные в контроллере можно связать с любой java-компонентой. Возможны различные варианты связок методов контроллера с java-компонентами, т.е. один метод может прослушивать несколько java-компонент и одну java-компоненту может прослушивать несколько контроллеров. При возникновении события на java-компоненте, по порядку вызываются все методы, прослушивающие данную java-компоненту. Контроллер и GUI выполняются на стороне клиента. Система ГИС предоставляет множество java-компонент для реализации клиентского интерфейса, такие как кнопки, таблицы, лейблы, различные текстовые поля и редакторы, просмотрщик для PDF-документов, просмотрщик карты, поля для графиков, поля для просмотра времени и иные компоненты для управления и работы с данными. С помощью специальных плагинов в среде Eclipse возможен просмотр и редактирование графического интерфейса приложения. В редакторе графического интерфейса возможно устанавливать любое расположение предоставляемых java-компонент. Так же в редакторе производится установка всех свойств выбранных компонент. Например, таких как возможность редактирования полей, названия лейблов, возможность фокусировки и т.д. Кроме того система ГИС поддерживает создание логических условий. Логические условия чаще всего применяются для того, чтобы запретить или разрешить редактирование полей, сделать их фокусируемыми или запретить фокусировку или разрешить или запретить различные действия на пользовательском интерфейсе. Логические условия вычисляются динамически при возникновении различных событий на пользовательском интерфейсе, результат их вычисления зависит от содержимого полей данного приложения.

Контролер в свою очередь связан с *сервисом*. Данная связь происходит посредством сети. Сервис является связующим звеном между клиентской частью и сервером. Сервис так же реализован на языке Java и выполняется на отдельном сервере с помощью программного продукта IBM WS Application Server.

Данный продукт является промежуточным программным обеспечением, которое позволяет приложениям работать на разных платформах на основе веб-технологий. WS использует открытые стандарты XML[2].

С помощью сервиса производится отдельная часть реализации бизнес логики приложения. А именно производится проверка данных, введенных пользователем. Например, это может быть проверка допустимых комбинаций данных. Так же с помощью сервиса можно запустить другое клиентское приложение. При этом есть возможность автоматически заполнить в новом приложении различные поля, основываясь на данных из предыдущего приложения. Сервис представляет собой java-класс, в котором описаны различные методы. Методы сервиса удаленно вызывает контроллер.

Далее сервис вызывает серверную часть приложения с помощью специальных компонент называемых *бинами*. Бины представляют собой несколько java-классов и служат для связи между различными системами. Таким образом, возможно мульти платформенное программирование. В системе ГИС клиентская часть полностью реализована на языке Java, а серверная на языке Ген, поддерживаемом компанией СА.

### 1.2 Разработка компонент

Разработка сервера производится на языке Ген. Язык Ген является объектным и высоко структурированным. Т.е. в данном языке все является объектом и существует жесткая структура написания программ на данном языке. Структурированность данного языка помогает избегать синтаксических и большую часть логических ошибок. Для данного языка поддерживается специальная среда, которая помогает разработчикам в создании и управлении крупномасштабными мульти платформенными приложениями. С помощью данного языка происходит описание архитектуры приложения и последующая генерация структурного скелета приложения и его связующих звеньев. В случае системы ГИС производится описание контроллера, сервиса и бинов. Среда разработки использует гибкие методологии для дизайна, развертывания и управления приложениями. Т.е. с помощью данного языка и данной системы возможна перегенерация всей архитектуры приложения в любой момент разработки и адаптация процесса разработки к новым требованиям.

При развертывании на сервере программы, написанной на языке Ген – происходит преобразование исходного кода языка Ген в низкоуровневый код на языке COBOL, после чего выполняется компиляция, линковка и запуск программы на сервере[3]. Для развертывания сервера используется мэинфреим под управлением операционной системы для серверов z/OS, которую разрабатывает и поддерживает компания IBM.

Язык Ген является процедурным. Таким образом, в языке Ген программа описывается в виде процедуры, из процедуры могут вызываться блоки кода и другие процедуры. В процессе выполнения процедуры происходит работа с объектами. Так же стоит отметить, что у каждой процедуры есть *импорты* и *экспорты*. Импорты – те объекты, которые подаются процедуре на вход. Экспорты – те объекты, которые процедура возвращает.

Архитектура всего приложения разрабатывается на языке Ген с помощью процедур следующим образом. Изначально разрабатывается процедура, в которой описываются события соответствующие методам сервиса. В данных событиях происходит различная установка системных данных и, при необходимости вызова сервера, описывается, какой сервер должен быть вызван при возникновении данного события. Далее происходит описание процедуры, соответствующей контроллеру. В данной процедуре, для описания методов контроллера, так же используются события. В каждом событии контроллера указывается то, какие события из сервиса необходимо вызвать в процессе обработки. Так же устанавливается различная системная информация. Если возникает такая необходимость – можно из контроллера произвести вызов сервера, миновав сервис. В экспортах данной процедуры происходит определения объектов, которые будут пересылаться между клиентской частью и сервером, а в импортах – описание объектов, которыми можно инициализировать данное приложение для оперативного открытия через сервис (вызов данного приложения из других приложений).

Система ГИС предоставляет специальный плагин, который производит чтение и анализ описанных выше процедур, написанных на языке Ген. Рассматриваемый плагин предназначен для генерации каркаса приложения, т.е. для создания структуры всех java-классов данного приложения. Например, таких как контроллер или сервис. Так же данный плагин предназначен для полной или частичной перегенерации архитектуры приложения. Данная способность позволяет поддерживать возможность разработки с применением гибких методологий, в которых требования к дизайну и архитектуре конечного продукта постоянно меняются.

Сам сервер описывается на языке Ген в виде процедуры. Из данной процедуры может происходить вызов отдельных блоков кода и других процедур, называемых *операциями* или *компонентами*. Блоки кода обычно используются для обобщения выполняемых действий, например, таких как управление данными (создание, модификация, удаление) или считывание данных. Операции предназначены для программной реализации работы с данными, т.е. непосредственного обращения к серверам с базами данных и выполнения SQL-запросов в базе данных DB2. Так же операции могут выполнять преобразование данных и возвращение определенных значений, например, текущее время или дата. При работе с базами данных используются специальные объекты-сущности, которые соответствуют таблицам базы данных. Архитектурно операции состоят из следующих блоков:

1. интерфейс;
2. маппер;
3. блоки копирования;
4. имплементация.

При вызове какой-либо компоненты используется интерфейс. Интерфейс разрабатывается следующим образом. Изначально в интерфейсе определяется, какие объекты принимает данная компонента и какие возвращает. А программная реализация состоит только из вызова маппера. После этого происходит компиляция и линковка интерфейса. Далее из интерфейса удаляется маппер, и новая версия интерфейса закачивается на сервер, но не происходит компиляции и линковки. Если не удалить маппер, то при импортировании компоненты в другие пакеты пришлось мы импортировать все блоки кода. А таким образом, если маппер удален, необходимо импортировать только пустой интерфейс. При этом остальные блоки кода содержатся только в исходном пакете. Однако при вызове интерфейса будет выполняться его последняя откомпилированная версия, в которой маппер содержится.

Маппер выполняет поочередный вызов блоков копирования и блоков имплементации. Блоки копирования выполняют преобразование обычных объектов в объекты-сущности, с помощью которых происходит обращение и работа с базой данных, и наоборот. А блок имплементации выполняет непосредственное обращение и работу с базой данных.

### 1.3 Разработка репортов

Система ГИС так же предоставляет возможность оперативного извлечения информации из базы данных и представления этой информации в удобном виде для оценки и наглядного восприятия. Для это служат отдельные приложения, называемые *репортами*. Разработка данных приложений так же производится с помощью среды Eclipse с использованием специальных плагинов.

Разработка репортов производится с помощью специального скриптового языка баттерфляй и SQL-запросов. Каждый репорт содержит основной скрипт, который выполняется при запуске репорта. Так же репорт может содержать неограниченное количество SQL-запросов. В процессе выполнения основного скрипта происходит вызов SQL-запросов в любой последовательности и постепенное заполнение выходных данных. Выходные данные основного скрипта обычно представляются в виде таблицы. Т.е. происходит постепенное (построчное) заполнение выходной таблицы. В процессе заполнения каждой строки у разработчика есть возможность производить различные операции с данными, выполнять их преобразование или же выполнять другие запросы аналогичным способом. Таким образом, у разработчика появляется гибкий инструмент для выборки данных из таблицы, при использовании которого появляется возможность эффективно использовать SQL-подзапросы[4].

Перед выполнением запросов с помощью скриптового языка есть возможность произвести проверку введенных данных на разрешенные значения или ошибки.

При проверке и обработке данных разработчику предоставляются удобные библиотеки для модификации и обработки данных. Система ГИС располагает обширными библиотеками для работы с:

1. числами;
2. датами и временем;
3. строками;
4. SQL-запросами;
5. групповой обработки данных.

Данные для репорта вводятся с помощью специального диалогового окна, которое отображается при запуске скрипта.

После выполнения основного кода результат записывается в XML-файл. При необходимости произвести специфическую обработку данных, например нетривиальную сортировку, полученный XML-файл обрабатывается с помощью еще одного преобразующего скрипта, написанного на рассматриваемом языке. После всех обработок данные выдаются в заранее заданном виде. Для отображения данных могут быть созданы шаблоны следующих форматов XML, EXCEL (XLS, XLSX), PDF. Один репорт может содержать несколько форматов вывода, и каждый формат может содержать свой преобразующий скрипт. При этом при создании шаблонов, возможно использование всех функций и преимуществ выбранного формата. Например, для EXCEL документов с помощью макросов можно произвести построение “на лету” различных графиков, диаграмм, PivotTables и т.д. с выбранными данными.

### 1.4 Разрабатываемая система

Конечной целью в данной работе была выбрана логистическая система для морского транспорта. Данной системе необходимо предоставлять такой функционал как отслеживание кораблей и выдачу истории о задержках. Прогнозирование того факта, насколько благоприятно в дальнейшем будут доставляться грузы, если политика доставки не будет изменяться. Исследование влияния неблагоприятных задержек на общую картину стоимости всей доставки. Т.е. основной целью данной системы является оптимизация доставки грузов и уменьшение непредсказуемых ситуаций, которые негативно сказываются на общей картине.

Существуют аналоги транспортных систем. Стоит отметить, что большинство программного обеспечения в этой сфере коммерческое и не существует в свободном доступе. Самым близким аналогом является известная программа 1С, в которой присутствует планирование перевозок. Вид программы представлен на рисунке 1.1. Данная версия может оперировать и с железнодорожным транспортом и с судами, поскольку алгоритмическая часть в обоих случаях очень похожая. Программа 1С в данной версии обладает следующим функционалом: консолидация заказов по одному направлению; планирование перевозки с учетом маршрута, транзитного времени и операций на расположениях; подбор и разбивка по вместимости подвижного состава; расчет стоимости перевозки в соответствии с актуальным тарифом[5]. Известно, что данная программа является платной и не распространяется как свободное программное обеспечение. Кроме того, в отличие от разрабатываемой системы, в данной программе не производится анализ предыдущих доставок с целью улучшения качества обслуживания, минимизации задержек и выявления проблемных участков.

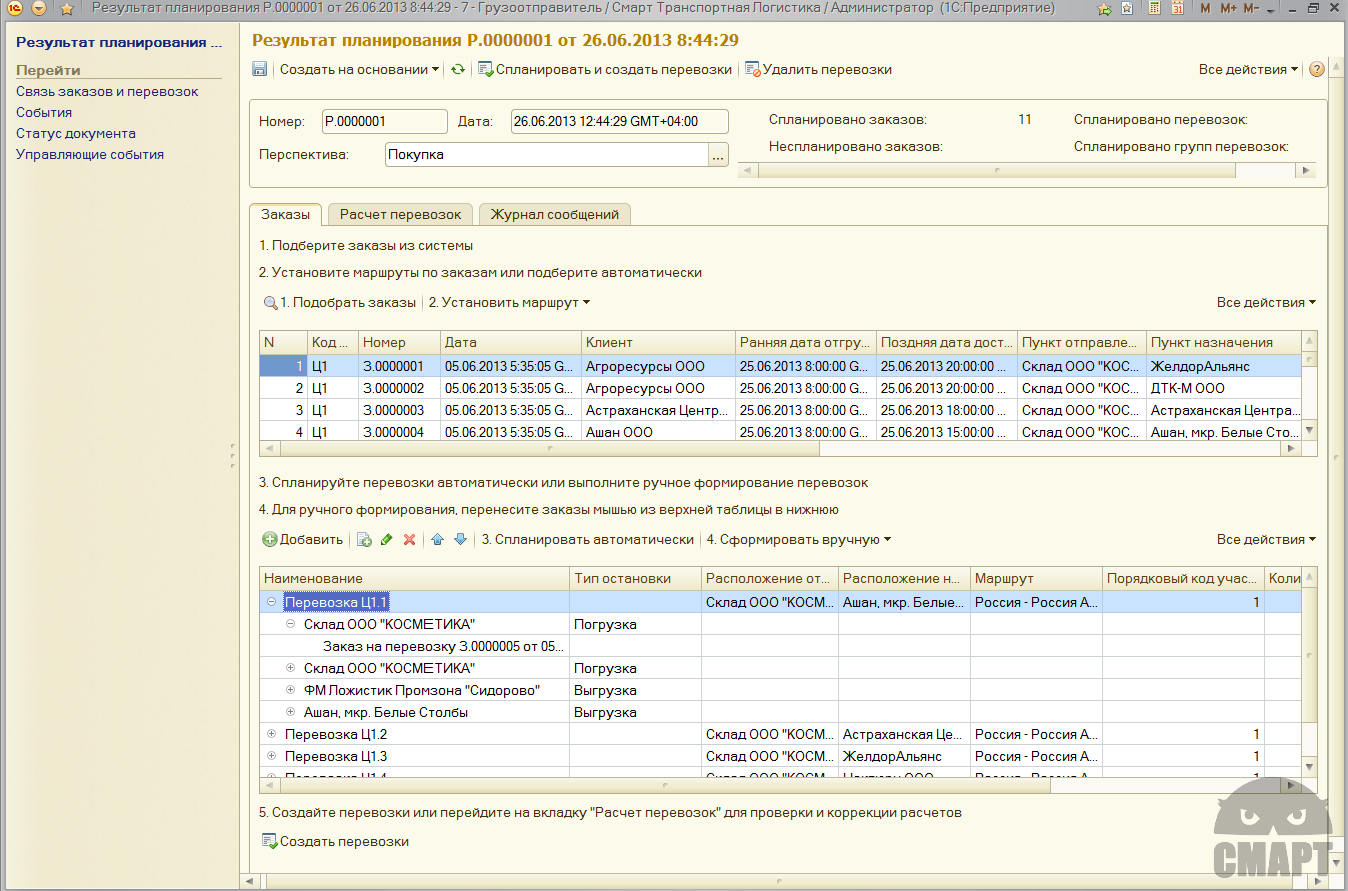


Рисунок . – пример программы аналога 1С

Большинство программ в данной сфере оперируют только с автомобильным транспортом. В пример можно привести популярную программу Open Door Logistic Studio. Она пользуется популярностью из-за широкого функционала, кроме того, это программное обеспечение распространяется бесплатно в случае некоммерческого использования. Программа Open Door Logistic Studio позволяет распределить грузы по имеющемуся транспорту и отобразить это на карте, пример представлен на рисунке 1.2. Кроме того, возможно мануально изменить порядок доставки и перебросить точки доставки с одного транспорта на другой. Так же есть возможность выбрать сектора, за которые ответственны исходные точки, т.е. точки отправки. У данной программы присутствуют все аналогичные отличия, как и у 1С, кроме того, нет возможности работать с железнодорожным и морским транспортом.

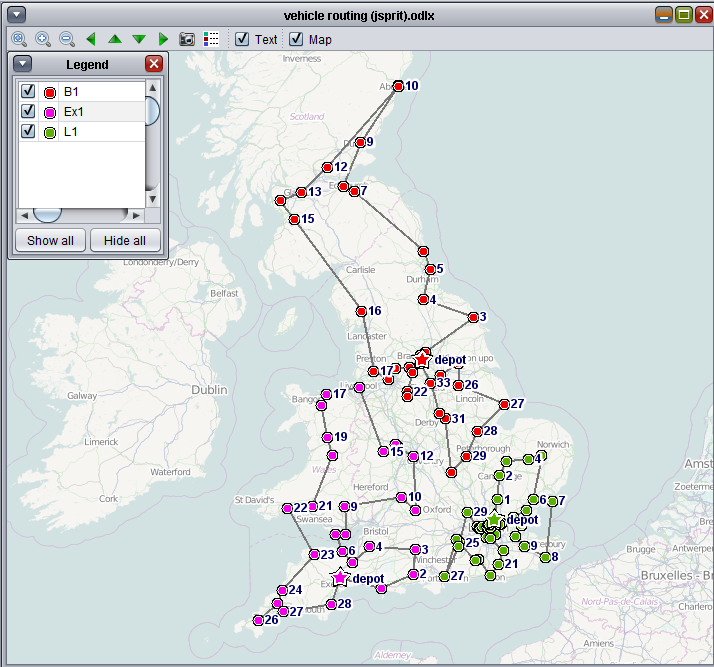


Рисунок . – пример программы аналога Open Door Logistic Studio

Таким образом, основные отличия от аналогов приведены в таблице 1.1.

Таблица 1.1 – отличия аналогов от разрабатываемой системы

|  |  |  |
| --- | --- | --- |
| Отличие | Разрабатываемая система | Большинство аналогов |
| Представление данные | Удобное представление данных. В т. ч. вывод на карту. | Удобное представление данных. В т. ч. вывод на карту. |
| Виды транспорта и типы грузов | Судовой морской транспорт. Каждый груз индивидуальный. | Преимущественно дорожный транспорт, автомобили. Грузы однотипные (кроме системы 1С). |
| Связь грузов и транспорта | Груз привязывается к маршруту транспорта. Доставка с перегрузками. | Маршрут транспорта планируется в зависимости от грузов. Доставка без перегрузок. |
| Анализ истории доставок | Возможен анализ предыдущих задержек, прогнозирование будущих на основе предыдущих. | Анализ не предусмотрен, возможен только с помощью сторонних программ или группой экспертов. |
| Планирование доставки | Планирование на основе расписания транспорта, начальных и конечных точек грузов, на основе проанализированной истории задержек кораблей. | Планирование на основе начальных и конечных точек грузов. Система 1С для ЖД транспорта – на основе расписания транспорта. История доставок не учитывается. |

**ГЛАВА 2**

# МОДЕЛИ И МЕТОДЫ ПРОГНОЗИРОВАНИЯ

### 2.1 Основные модели и методы прогнозирования

Модель прогнозирования – некоторое представление объекта прогнозирования, которое описывает исследуемый процесс. Исследование модели прогнозирования позволяет получить информацию о возможных состояниях объекта в будущем.

Метод прогнозирования – способ исследования объекта прогнозирования, направленный на разработку прогноза. Методы прогнозирования состоят из последовательности действий, которые необходимо совершить для получения той или иной модели прогнозирования.

Методы прогнозирования делятся на две группы: интуитивные и формализованные. В интуитивных методах используются суждения и оценки экспертов. Сферы применения интуитивных методов: маркетинг, экономика, политика и т.п. Таким образом, прогнозирование интуитивными методами используется либо в сферах, которые очень сложно описать математически, либо в очень простых сферах, когда нет необходимости в описании с помощью математической модели. В обратном случае применяются формализованные методы. При применении формализованных методов, строятся модели прогнозирования. Т.е. происходит определение некоторой математической зависимости, с помощью которой возможно вычисление будущего значения исследуемого процесса.

Формализованные методы включают в себя две группы моделей: модели предметной области и модели временных рядов.

Модели предметной области относятся к конкретной сфере применения, в которой существуют специфические уравнения или законы, которые необходимо учитывать для рассмотрения и исследования данной области. Примером можно привести прогноз погоды, в нем используются уравнения термодинамики и динамики жидкостей.

Модели временных рядов – это математические модели прогнозирования, с помощью которых возможно нахождения зависимости будущего значения исследуемого процесса от прошлых значений. На основе этой зависимости производится вычисление прогноза. Модели временных рядов можно считать универсальными, поскольку их внешний вид не зависит от природы происхождения временного ряда и остается неизменным. Зависимость между будущим значением и прошлыми значениями выражается в виде аппроксимирующей функции.

В зависимости от области определения параметров аппроксимирующей функции выделяют две основные группы моделей: глобальные и локальные.

В глобальных моделях параметры аппроксимирующей функции идентифицируются посредством использования всех известных значений ряда. Основная цель глобальных моделей – получение глобальных характеристик системы. К ним относятся: авторегрессионные модели, сингулярный спектральный анализ.

Локальные модели основаны на принципе локальной аппроксимации (LA). Применяются в целях прогнозирования динамики ряда[6].

Стоит отметить, что на сегодняшний день для прогнозирования временных рядов имеется огромное количество моделей и методов для построения прогноза, например, таких как SVM модели, модели на основе генетических алгоритмов и многие другие.

### 2.2 Сингулярный спектральный анализ

Метод сингулярного спектрального анализа (SSA) используется для определения основных составляющих временного ряда и подавления шума. Метод SSA позволяет:

1. различать составляющие временного ряда, полученные из последовательности значений какой-либо величины;
2. находить заранее неизвестные периодичности ряда;
3. сглаживать исходные данные на основе отобранных составляющих;
4. наилучшим образом выделять компоненту с заранее известным периодом;
5. предсказывать дальнейшее поведение наблюдаемой зависимости.

В основе SSA лежит построение множества векторов задержек:

|  |  |
| --- | --- |
|  | (2.1) |

Метод задержек устанавливает переход от исходного одномерного (скалярного) временного ряда к многомерному (векторному) представлению. Каждый многомерный вектор образуется из некоторого числа p следующих друг за другом значений исходного временного ряда:

|  |  |
| --- | --- |
|  | (2.2) |

Здесь каждая квадратная скобка — вектор в p–мерном пространстве задержек; последовательность таких векторов задает матрицу задержек , где *N* — число элементов исходного ряда.

Особенностью SSA является обработка матрицы X по алгоритму, похожему на метод главных компонент[7]. Суть метода главных компонент состоит в снижении размерности исходного пространства факторов (задержек) с помощью ортогонального линейного преобразования. Полученные таким образом новые переменные и называют главными компонентами. Применение этого метода позволяет сгладить исходный ряд, снизить уровень случайных возмущений, повысить отношение сигнал/шум.

Способы прогнозирования на базе SSA следующие: (а) “Гусеница” и (б) метод авторегрессии, который применяется по отдельности к каждой из выбранных компонент разложения. Ниже представлены основные этапы применения SSA к конкретному ряду .

Изначально происходит преобразование одномерного ряда в многомерный ряд. Для такого преобразования необходимо взять некоторое число задержек .

|  |  |
| --- | --- |
|  | (2.3) |

|  |  |
| --- | --- |
|  | (2.4) |

Построенная матрица X является прямоугольной, но в предельном случае, т.е. при τ = (N + 1)/2 и нечетном N, она вырождается в квадратную.

Далее происходит построение для матрицы X соответствующей ковариационной матрицы:

|  |  |
| --- | --- |
|  | (2.5) |

После чего происходит определение собственных значений и собственных векторов матрицы *С*. Для этого ее необходимо разложить по собственным векторам:

|  |  |
| --- | --- |
|  | (2.6) |

Где введены следующие матрицы: диагональная матрица собственных чисел Λ и ортогональная матрица собственных векторов матрицы C.

|  |  |
| --- | --- |
|  | (2.7) |

|  |  |
| --- | --- |
|  | (2.8) |

Таким образом, будут выполняться следующие равенства:

|  |  |
| --- | --- |
|  | (2.9) |

|  |  |
| --- | --- |
|  | (2.10) |

последнее равенство справедливо только в случае предварительно нормированных строк матрицы X.

Далее происходит переход к главным компонентам. Матрицу собственных векторов V обычно представляют как матрицу перехода к главным компонентам исходного ряда.

|  |  |
| --- | --- |
|  | (2.11) |

Где величины Yi , i = 1, 2, . . . , τ , — строки длины n. При этом собственные значения можно рассматривать как вклад главных компонент в общее информационное содержание временного ряда . Тогда по полученным главным компонентам можно полностью восстановить исходную матрицу

|  |  |
| --- | --- |
|  | (2.12) |

а по ней в свою очередь можно восстановить временной ряд .

Преобразования , i = 1, 2, . . . , τ , , *l* = 1, 2, . . . , *n*, являются *линейными фильтрами*.

Каждый *i*-й собственный вектор включает в себя *τ* компонент, т.е. . Построим зависимость значений компонент , k = 1, 2, . . . , *τ* , от их номера: . Тогда, используя свойство ортогональности собственных векторов, дальнейший анализ последовательности можно провести, изучая диаграммы, построенные по аналогии с *фигурами Лиссажу*. Именно по осям попарно откладываются компоненты , . Если построенные диаграммы оказываются близкими по форме к окружности, то функции , будут подобны периодическим с близкими амплитудами и со сдвигом фазы около четверти периода.

Таким образом, для некоторых пар собственных векторов Vi , Vj можно вычислить величину, имеющую смысл периода. Следовательно, графический анализ дает представление о частотах компонент, формирующих исходный временной ряд .

Далее возможно произвести SSA-сглаживание. Предположим, что из τ компонент для дальнейшего анализа оставлены лишь первые *r*. Тогда для восстановления исходной матрицы X используются первые *r* собственных векторов . В таком случае

|  |  |
| --- | --- |
|  | (2.13) |

где – восстановленная матрица, имеющая *n* столбцов и τ строк. Теперь исходный временной ряд, воссозданный из этой матрицы, определится как

|  |  |
| --- | --- |
|  | (2.14) |

Данный способ получения последовательности называется SSA-сглаживанием исходного временного ряда по первым *r* компонентам из τ .

### 2.3 Прогноз с помощью метода SSA

На этом этапе строится ряд . При этом прогноз на p точек вперед осуществляется как применение p раз операции прогноза на одну точку.

Базовая идея нахождения значения состоит в следующем. Пусть имеется набор значений , , … , . Теперь построим выборку в виде матрицы X. В качестве базиса поверхности, содержащей эту выборку, можно взять отобранные ранее собственные векторы , , . . . , матрицы C.

Запишем параметрическое уравнение этой поверхности как , где каждому значению вектора *S(P)*, представляющего собой столбец высотой τ , соответствует набор из *r* параметров . Тогда *k*-му (*k* = 1, 2, . . . , *n*) столбцу исходной матрицы X соответствует свой набор значений параметров = ( , , . . . , ) и, следовательно,

|  |  |
| --- | --- |
|  | (2.15) |

Для предсказания значения необходимо построить (*n + 1*)-й столбец матрицы X, которому соответствует некоторое значение параметров . Этот набор параметров можно найти из соотношения , исходя лишь из данных . При этом прогнозируемый столбец будет записан следующим образом: .

Введем следующие обозначения:

|  |  |
| --- | --- |
|  | (2.16) |

|  |  |
| --- | --- |
|  | (2.17) |
|  |  |

Параметры можно определить из системы уравнений относительно . Таким образом, окончательное выражение для прогнозируемого значения будет выглядеть как

|  |  |
| --- | --- |
|  | (2.18) |

Для предсказания следующих значений в простейшем случае требуется лишь изменить соответствующим образом матрицу *Q* и вновь умножить ее на величину / ( ). Однако дополнительно при этом можно для каждой следующей точки повторять частично или полностью весь алгоритм метода SSA. В этом случае будут также меняться матрицы и .

**ГЛАВА 3**

# ГЕНЕТИЧЕСКИЕ АЛГОРИТМЫ

### 3.1 Описание генетических алгоритмов

Генетический алгоритм (англ. genetic algorithm) — это эвристический алгоритм поиска, используемый для решения задач оптимизации и моделирования путём случайного подбора, комбинирования и вариации искомых параметров с использованием механизмов, аналогичных естественному отбору в природе. На сегодняшний день генетические алгоритмы нашли применения во многих и различных сферах. Генетические алгоритмы применяются для решения следующих задач:

1. оптимизация функций;
2. оптимизация запросов в базах данных;
3. разнообразные задачи на графах (задача коммивояжера, раскраска, нахождение паросочетаний);
4. настройка и обучение искусственной нейронной сети;
5. задачи компоновки;
6. составление расписаний;
7. игровые стратегии;
8. теория приближений;
9. искусственная жизнь;
10. биоинформатика (фолдинг белков);
11. синтез конечных автоматов;
12. настройка ПИД регуляторов.

Генетические алгоритмы - адаптивные методы поиска. Они основаны на генетических процессах биологических организмов: биологические популяции развиваются в течение нескольких поколений, подчиняясь законам естественного отбора. Генетические алгоритмы основаны на принципе "выживает наиболее приспособленный", который сформировал Чарльз Дарвин. Подражая этому процессу генетические алгоритмы способны "развивать" решения реальных задач, если те соответствующим образом закодированы.

Для решения некоторой прикладной задачи с помощью генетического алгоритма её необходимо изначально правильно задать и формализовать. Основным принципом такой формализации является представление решения задачи в виде вектора определенных объектов. Такой вектор называется *генотипом* и состоит он из элементов, называемых *генами.* Каждый ген может представлять собой бит, число или некоторый другой объект. Если рассматривать генетический алгоритм в классической реализации, то необходимо чтобы все генотипы имели одинаковую длину, однако, существуют ситуации, когда практические задачи требуют выхода за эти рамки. Поэтому на практике этим часто пренебрегают и создают вариации генетических алгоритмов, в которых нет жестких ограничений, накладываемых на размерности генотипов.

Изначально создается множество генотипов начальной популяции. Это множество обычно создается случайным образом. Целью данного множества не является найти оптимальное решение, а всего лишь создать некоторую популяцию, которая будет подходить для описания задачи, и которую можно будет развивать в последующих действиях. Поэтому случайное создание популяции вполне подходит для этих целей. Далее все популяции оцениваются с помощью *функции приспособленности*, или как её еще называют, *фитнесс функции*. В результате чего с каждым генотипом есть возможность ассоциировать некоторую величину, называемую *приспособленность*, которая определяет, насколько хорошо рассматриваемый генотип решает текущую задачу.

Таким образом, получается множество решений, которое называется *поколение*. Далее из поколения выбираются некоторые решения, к которым впоследствии будут применены генетические операторы. Существуют различные способы выбора этих генов, зачастую более высокую вероятность быть выбранными имеют гены с наилучшими значениями функции приспособленности. Однако выбираются не всегда только лучшие гены, необходимо чтобы каждый ген имел некоторую вероятность быть скрещенным в некоторым другим. Далее генам применяются генетические операторы, в большинстве случаев это *скрещивание* и *мутация*. После выполнения этих операторов образуются новые решение, т.е. новые гены, для которых необходимо произвести вычисление значения функции приспособленности. После проделанных операций производится отбор, или как его еще называют *селекция*, лучших решений в следующее поколение[8].

Данные операции повторяются итеративно, таким образом, производится моделирование эволюционного процесса. Такой процесс должен продолжаться несколько жизненных циклов, т.е. несколько поколений. Для остановки такого процесса определяется специальный соответствующий критерий. Возможны различные варианты этого критерия, так же можно строить различные комбинации вариантов. В качестве примера можно привести следующие критерии остановки жизненного цикла:

1. нахождение глобального решения;
2. нахождение решения с заранее заданной погрешностью;
3. ограничение количества поколений для эволюционного процесса;
4. ограничение количества генов, которые будут создаваться в процессе эволюции;
5. ограничение времени, отведенного на эволюцию.

Обычно генетические алгоритмы используются для нахождения решений в многомерных пространствах поиска.

Таким образом, можно выделить основные этапы, которые необходимо произвести при решении задачи с помощью генетического алгоритма:

1. задать функцию приспособленности, или фитнесс функцию;
2. создать начальную популяцию;
3. произвести операцию размножения (скрещивания);
4. произвести операцию мутации;
5. вычислить значение приспособленности с помощью фитнесс функции для новых генов;
6. сформировать новое поколение (произвести операцию селекции);
7. проверить условия остановки, если выполняются, то остановить алгоритм и выбрать ген с наилучшим значением функции приспособленности, иначе перейти к пункту 3.

Схематично генетический алгоритм представлен на рисунке 3.1.

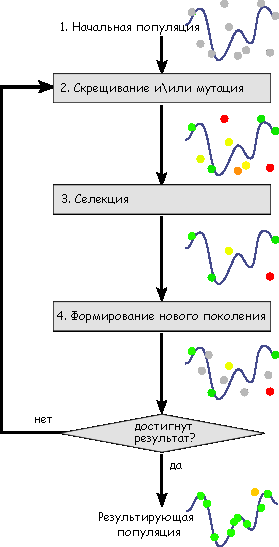


Рисунок . – схематичное представление генетического алгоритма

### 3.2 Начальная популяция

Изначально, перед началом всего жизненного цикла необходимо создать начальную популяцию. Обычно разработке алгоритмов для создания начальной популяции не уделяют большое количество времени. В большинстве случаев для создания генов выбирают случайные значения. Так происходит в силу того, что на дальнейших шагах алгоритма будет постоянно происходить улучшение приспособленности популяции. Таким образом, начальная популяция может быть абсолютно неконкурентоспособной, у которой гены имеют низкие значения функции приспособленности. Главным жестким требованием для начальной популяции является то, чтобы все гены строго соответствовали формату особей популяции, и для них можно было бы применить функцию приспособленности. Итогом данного шага является популяция, содержащее некоторое количество особей.

### 3.3 Операция селекции

На этапе отбора особей для следующей популяции выполняется операция селекции. Выбрать определенную долю из всех особей, которая на данном этапе эволюции останется “выжившей”, является задачей операции селекции. Для того, чтобы реализовать операцию селекции существуют различные подходы. Основным принципом выборки генов в операции селекции является значение их функции приспособленности. Вероятность выжить у особи с лучшим значением функции приспособленности должна быть выше, чем у особи с худшим значением приспособленности. Доля выживших особей может быть как динамическим значением, так и параметром. По итогам выполнения операции селекции, остаются в живых sN особей из N исходных, где s – доля, которая должна выжить, выраженная в процентном соотношении. Остальные особи погибают.

Существуем множества подходов по реализации операции селекции, следующие подходы можно привести как пример:

1. Турнир – случайным образом выбирается некоторое количество особей, из них остается одна с наилучшим значением функции приспособленности. Зачастую выбирают две особи для участия в турнире.
2. Рулетка – для каждой особи подсчитывают вероятность быть выбранной на основе значений функций приспособленности. Вероятность определяется с помощью формулы , – где искомые вероятности, – значения функции приспособленности. Сумма всех вероятностей равна 1.
3. Метод ранжирования – для особи вероятность быть выбранной оценивается по месту с списке всех особей, отсортированному по значению функции приспособленности.

|  |  |
| --- | --- |
|  | (3.1) |

Гдe , b = 2 – a

### 3.4 Выбор генов, операция скрещивания и операция мутации

Для того чтобы произвести потомство, производится выбор генов, которые будут участвовать в скрещивании. Для скрещивания обычно выбирают двух родителей.

Для выбора этих родителей применяются следующие подходы:

1. оба родителя выбираются случайным образом;
2. первый родитель выбирается случайно, а второй берется максимально схожий с текущим выбранным родителем;
3. первый родитель выбирается случайно, а второй берется максимально отличным от текущего выбранного родителя.

Для второго и третьего пунктов можно применить два подхода по определению схожести. Первый подход заключается в определении близости значений функции приспособленности. Второй подход строится на анализе отличий в самих генах.

Операция скрещивания реализуется с учетом специфики каждой задачи в отдельности. На операцию скрещивания накладывается свойство, которое должно быть учтено при реализации. А именно то, что новых потомок должен наследовать черты обоих родителей некоторым образом смешав их. Условная схема операции скрещивания представлена на рисунке

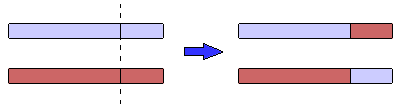


Рисунок . – схема операции скрещивания

Операция мутации так же специфична для каждой отдельно взятой задачи. При выполнении операции мутации выбирается некоторое количество особей. Далее в гены этих особей вносятся изменения случайным способом или с помощью применения некоторых алгоритмов. После данной операции получаются новые особи, для которых необходимо рассчитать величину функции приспособленности. Например, для случая, когда вектор представлен в виде бит и когда мутация распространяется только на один бит, мутацию можно представить в виде, отображенном на рисунке 3.3

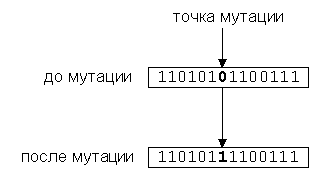


Рисунок . – схема мутации, пример

Основной целью применения операции мутации является ввод в популяцию особей с новыми свойствами. Данная характеристика помогает разнообразить популяцию и ввести в нее целый ряд новых генных свойств, которые могут быть унаследованы впоследствии. Эти характеристики операции мутации позволяют выходить из локальных максимумов при рассмотрении решений. Решением является глобальный максимум.

### 3.5 Применение генетических алгоритмов к поставленной задаче

Генетические алгоритмы являются, прежде всего, методом поиска оптимального решения. Одним из требований для разрабатываемой системы является поиск оптимального распределения грузов. Данную задачу можно представить в виде, который удовлетворяет требованиям генетических алгоритмов. Множество всех решений – множество всех распределений грузов, среди которых есть искомый элемент – оптимальный. Следовательно, геном является определенное распределение грузов по кораблям. Операции скрещивания и мутации, на основе существующих путей для грузов, должны производить новые пути, наследуя характеристики предыдущих. Операция фитнесс функции предназначена для вычисления стоимости заданного распределения. Начальную популяцию достаточно сформировать случайным способом. А операция селекции не требует исполнения каких-либо преобразований. Таким образом, появляется возможность реализации генетического алгоритма для поставленной задачи.

**ГЛАВА 4**

# ПОСТАНОВКА ЗАДАЧИ И РЕАЛИЗАЦИЯ

В качестве проекта было решено реализовать транспортную логистическую систему для морского транспорта, которая бы позволяла отображать положение кораблей, отображать перегрузки грузов, а так же отображать и оценивать задержки в пути следования кораблей.

Оценку задержек было решено производить при помощи алгоритма SSA, на сегодняшний день он является одним из самых популярных алгоритмов в прогнозировании. Для реализации транспортной логистической системы была выбрана система ГИС. Данная система предоставляет удобный и гибкий функционал для создания приложений. Клиентская часть реализована на языке Java, сервер реализован на языке Ген.

### 4.1 Модель данных

Для реализации данной логистической системы было принято участие в создании модели данных. Для хранения данных о кораблях было принято решение создать модель данных следующим образом. В отдельной таблице хранится информация о корабле, в отдельной таблице хранится информация о местоположении кораблей. Связь реализуется один ко многим, таким образом, предоставляется возможность хранить историю следования корабля. Данная модель представлена на рисунке 4.1.

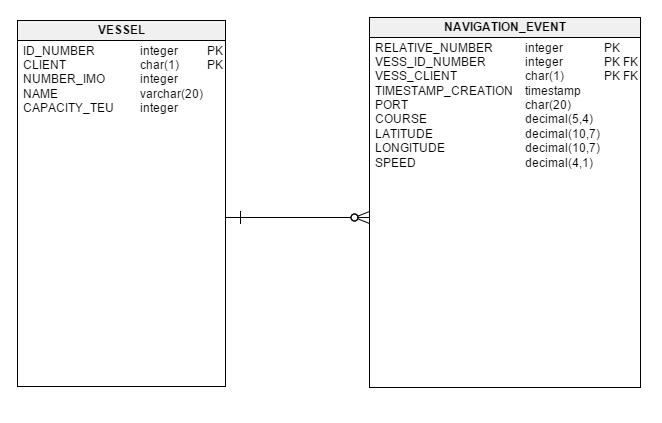


Рисунок . – модель данных “корабль-событие”

Для хранения данных о портах был выбран аналогичный подход. В отдельной таблице хранятся данные о терминале, в отдельной таблице хранятся данные о порте, в отдельной таблице хранятся данные о расположении. Таблицы порт и терминал связаны отношение один ко многим, т.к. в одном порту может быть несколько терминалов. Таблицы порт и местонахождение связаны один к одному. Данную модель данных можно увидеть на рисунке 4.2.

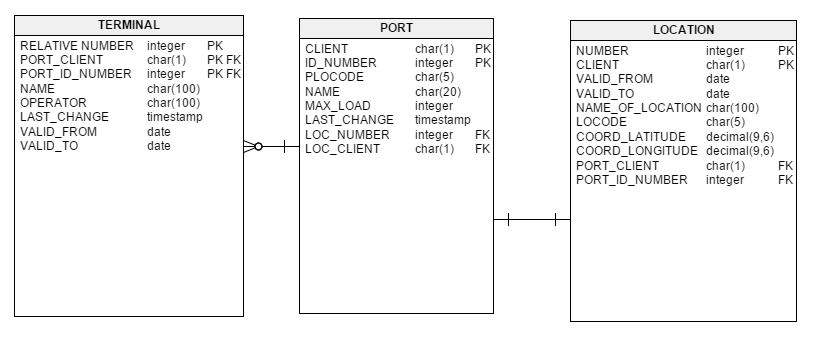


Рисунок . – модель данных “терминал-порт-местонахождение”

В данной работе так же отражена проблема задержек кораблей. В судоходстве необходимо оценивать задержки для предотвращения ошибок в расписании и запаздывании доставок грузов. Система ГИС поддерживает программы типа репорт, которые служат для оперативного предоставления информации. При проектировании репортов часто возникает необходимость учитывать новые специфические данные, для которых не предусмотрена архитектура базы данных. Например, некоторые вычисленные значения, или цвета для отображения уровней опасности. Тогда для ввода данных пришлось бы перепроектировать базу данных, но такой подход может оказаться неэффективным и трудозатратным. Поэтому, для вывода данных о задержках в отчет, было принято решение в отдельной таблице хранить список уникальных задержек, а данные об этих задержках хранить в отдельной таблице, при этом связь между таблицами установить один ко многим. Т.е. в таблице с данными будет два столбца характеризующих задержку, а именно: название атрибута и значение атрибута. А таблица с уникальными значениями будет содержать идентификационный номер задержки. Таким образом, возможно добавление все новой и новой полезной информации в базу данных относительно конкретной задержки. Данную модель можно увидеть на рисунке 4.3.

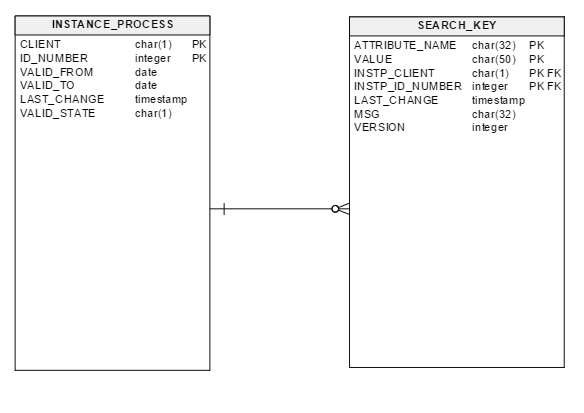


Рисунок . – модель данных “процесс-значение”

При перевозке груза из одного места в другое, обычно используется более одного корабля. Каждый корабль плывет по заранее заданному маршруту, при этом, приплывая в порт, происходит выгрузка некоторых привезенных грузов и загрузка новых грузов. Если же какой-нибудь из кораблей задерживается, то есть вероятность, что грузы, которые с него необходимо выгрузить, опоздают на загрузку на новый корабль. Таким образом, необходимо учитывать количество грузов, которые планируется загрузить на данный корабль и выгрузить с данного корабля. Каждому кораблю назначается свой маршрут. Маршруты, корабли и количество разгруженных и загруженных грузов хранится в разных таблицах. В таблице с количеством грузов, для большей эффективности, хранится порт, исходный маршрут и маршрут, на который грузы перегружаются. Данную модель можно увидеть на рисунке 4.4.

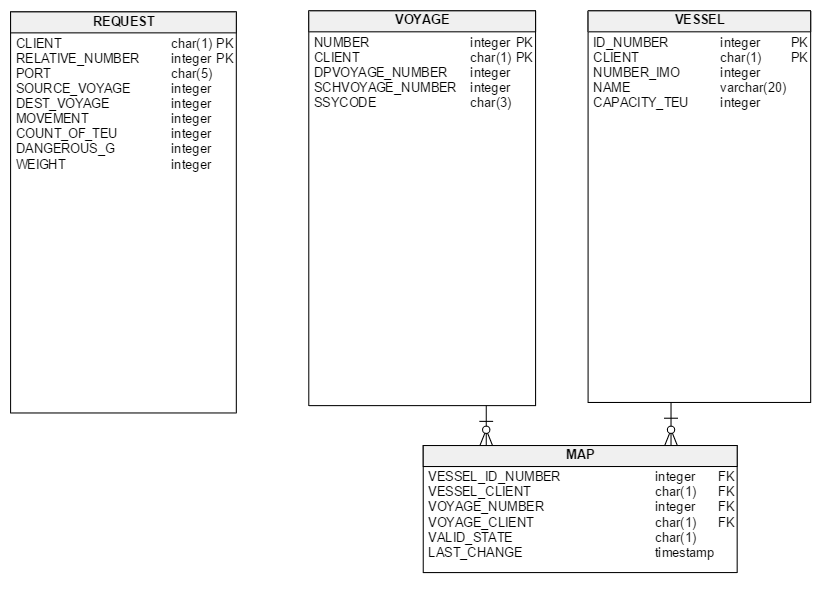


Рисунок . – модель данных “грузы-маршруты-корабли”

Модель данных реализована в базе данных DB2.

### 4.2 Реализованная система отслеживания кораблей

Для отображения положения кораблей, портов и терминалов было реализовано приложение, которое, с помощью доступного API, отображает карту, извлекает из базы данных информацию о местоположении интересующих кораблей, портов или терминалов и обозначает данное положение на карте соответствующими знаками. Карту предоставляет система ГИС. В построенной модели данных сохраняются широта и долгота положения всех перечисленных объектов (терминалы, порты и корабли). При этом так же хранится информация о каждом из объектов. После выборки этой информации из базы данных формируется XML-файл, и, с помощью соответствующего API, данный XML файл устанавливается в карту, предоставляемую системой ГИС. После всех действий система ГИС возвращает изображение карты, при этом в заданных положениях широты и долготы отображаются соответствующие объекты.

В реализованном приложении предоставляется возможность задания списка интересующих кораблей. Возможно задание идентификационного номера или имени корабля. Так же можно произвести поиск порта или терминала на карте. Поиск производится по введенным кодам портов, которые являются уникальными и хранятся в базе данных. Порты наносятся на карту аналогичным методом (рисунок 4.5).

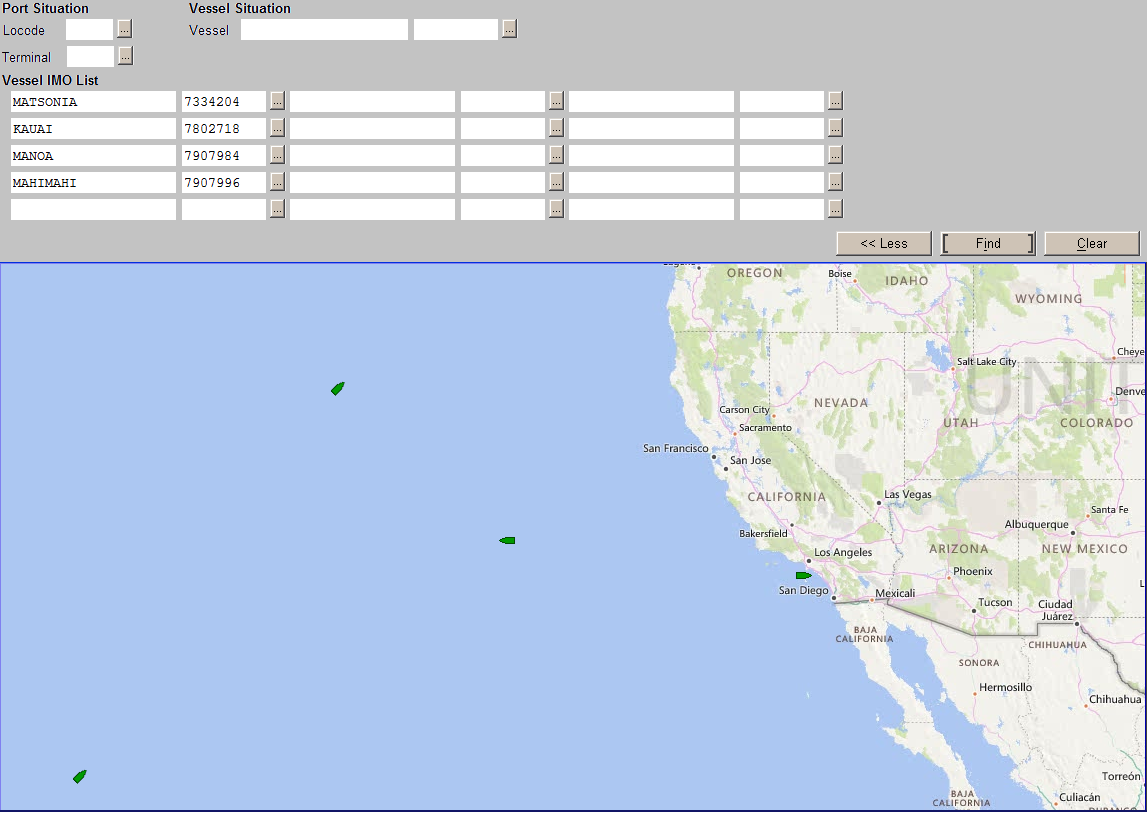


Рисунок . – приложение “карта”

Так же было реализовано приложение, которое отображает связь между кораблями. В данном приложении отображается запланированное количество грузов, которые необходимо перегрузить с одного корабля на другой. Соответствующее приложение помогает оценить, насколько корабли связаны друг с другом. Эти данные являются важными при оценке того, насколько велико влияние задержки корабля. Например, пусть данный корабль (корабль А) везет большое количество грузов, которые необходимо перегрузить на другой корабль (корабль Б). Пусть время t – время прибытия, корабля А в порт, а время s – время отправления корабля Б из этого порта. Таким образом чем ближе время t ко времени s – тем меньше остается времени на перегрузку и тем больше вероятность не успеть перегрузить все контейнеры с корабля А на корабль Б. Если же корабль А задержится и время t станет больше, чем время отплытия корабля Б. То контейнеры остаются в порту и ожидают следующего корабля, движущегося по маршруту, аналогичному тому, на который контейнер опоздал. Таким образом, информация о связности кораблей по контейнерам может быть ключевой при решении увеличить работу двигателей и прибыть в порт, минуя задержки. В реализованном приложении есть возможность ввести интересующий корабль, время его прибытия в порт и соответствующий порт. Или же интересующий идентификатор маршрута корабля и идентификатор порта в этом маршруте. После чего приложение отображает количество грузов, которые перегружаются с корабля, идущего по заданному маршруту на другие корабли и количество грузов которые загружаются на данный корабль и далее движутся по заданному маршруту (рисунок 4.6).

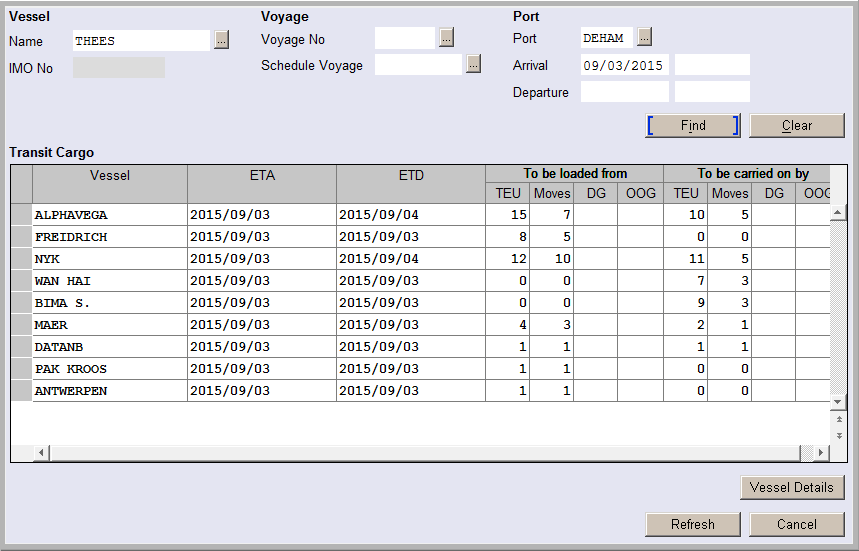


Рисунок . – приложение “связь между кораблями”

В реализованной системе находит отражение информация о задержках кораблей. Данная информация выводится с помощью программы типа репорт, реализованной с помощью средств и возможностей системы ГИС. При выполнении программы информация извлекается из разработанной модели данных. При запуске разработанной программы появляется диалоговое окно, в котором необходимо ввести данные для фильтрации информации, как представлено на рисунке 4.7.

Предоставляемые фильтры:

1. исходный регион – регион отправления корабля;
2. целевой регион – регион прибытия корабля;
3. направление движения;
4. нижняя граница времени отправления;
5. верхняя граница времени отправления;
6. нижняя граница времени прибытия;
7. верхняя граница времени прибытия.

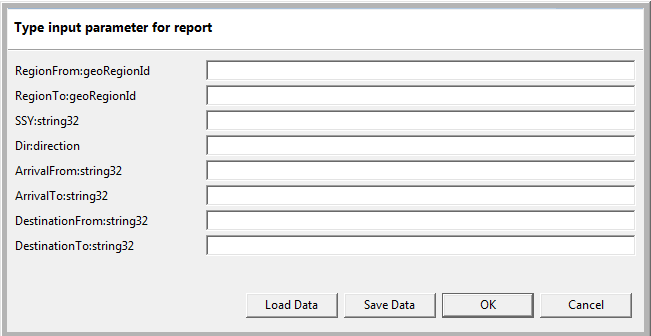


Рисунок . – приложение “репорт”, окно ввода параметров

После выполнения работы программы генерируется EXCEL документ, где отображена информация о задержках кораблей. Так же данный документ отражает информацию, которая позволяет произвести оценку для задержек. В документе указывается корабль, у которого возникла задержка и на какой части следования судна данная задержка возникла. Так же указывается величина этой задержки и когда именно произошла данная задержка. Таким образом, отображается информация, которую можно использовать для анализа и выявления проблемных мест в маршрутах (рисунок 4.8).

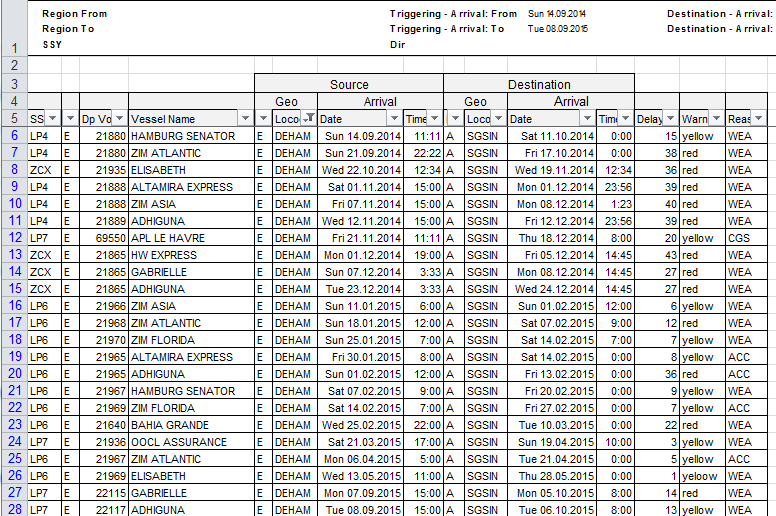


Рисунок . – результат работы приложения “репорт”

Задержки возникают очень часто, как показывает практика. Кроме того, задержки могут оказывать существенное влияние на работу всей системы. Поэтому, для более эффективного использования кораблей, необходимо производить анализ предыдущих значений задержек. Например, часто задержки возникают из-за погодных условий, которые подвержены периодичности. Алгоритм SSA на сегодняшний день является популярным алгоритмом для прогноза значений и применяется во множестве сфер. В данной системе был реализован алгоритм SSA для прогнозирования задержек.

Для прогнозирования были доступны данные о задержках для кораблей, следующих из Гамбурга в Сингапур. По расписанию, отправление по данному маршруту происходило 4 раза в месяц. Результат работы алгоритма представлен на рисунке 4.9.

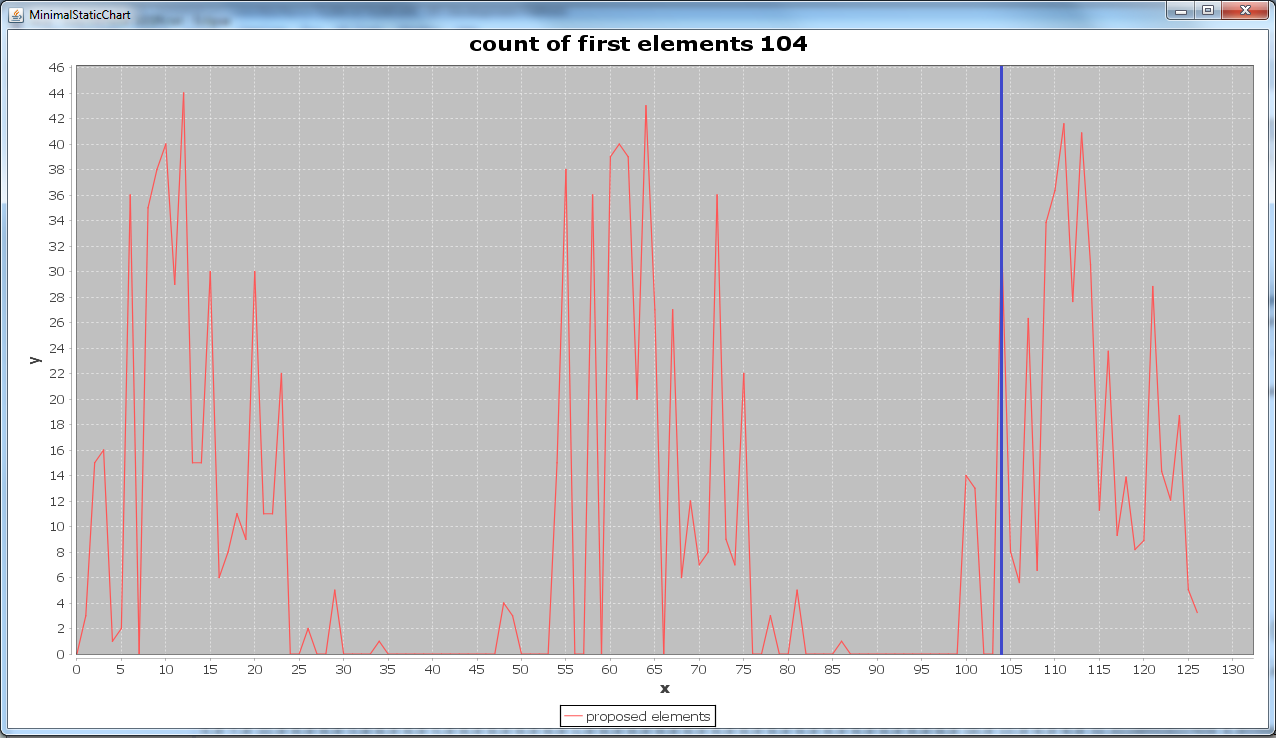


Рисунок . – прогноз задержек на зимний период

Таким образом, данная программа спроектировала будущие задержки на зимний период. Т.е. такие задержки ожидаются, если компания будет придерживаться такой же политики доставки грузов. А именно произойдет последовательность задержек, представленная в таблице 4.1. Из построенного графика и таблицы видно, что в зимний период ожидаются гораздо большие задержки, чем в летний период. Так же, на основании предыдущих задержек можно выявить, в какие месяцы задержки будут наиболее сильными. Например, для данной части маршрута, наибольшие задержки случаются в ноябре и декабре. В январе же, задержки ожидаются в два раза меньше, чем в предыдущие два месяца. Таким образом, если связь между кораблями по грузам высокая, то необходимо принять решение об увеличении работы двигателей в эти месяцы.

Таблица 4.1 – спроектированная последовательность

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Октябрь | | | | ноябрь | | | | декабрь | | | | январь | | | | февраль | | | | ... | |
| 8 | 5 | 26 | 6 | 33 | 36 | 41 | 27 | 40 | 30 | 11 | 23 | 9 | 13 | 8 | 8 | 28 | 14 | 12 | 18 | 5 | 3 |

Так же производилось сравнение уже известных задержек с тем, что выдает алгоритм. Данное сравнение представлено в таблице 4.2 и на рисунке 4.10. При оценке данной таблицы можно судить о том, что в летние месяцы ожидаются небольшие задержки в границах до 4-х часов, но уже в сентябре и октябре величины задержек станут значительно увеличиваться.

Таблица 4.2 – спроектированная последовательность

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Март | | | | Апрель | | | | Май | | | | Июнь | | | |
| Прогноз | 0 | 0 | 1 | 4 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Реальные значения | 0 | 0 | 3 | 0 | 0 | 5 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
|  | Июль | | | | Август | | | | Сентябрь | | | | Октябрь | | | |
| Прогноз | 0 | 0 | 0 | 0 | 0 | 0 | 4 | 2 | 0 | 0 | 3 | 21 | 31 | 5 | ... | |
| Реальные значения | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 14 | 13 | 0 | 0 | 15 | 29 | ... | |

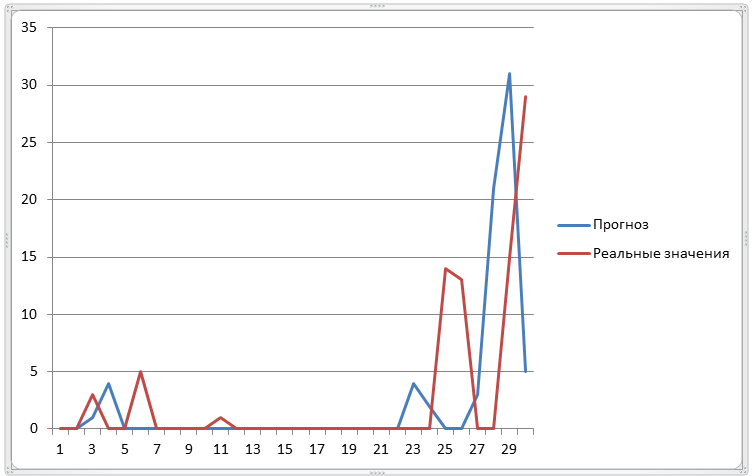


Рисунок . – сравнение прогноза с реальными значениями

Для тестирования реализованного алгоритма так же была использована база чисел Вольфа. Числа Вольфа – числа, характеризующие солнечную активность и количество пятен на солнце. Числа Вольфа измеряются каждый год и хранятся в общедоступной базе. Выявлено, что числа Вольфа носят периодический характер, следовательно, на них можно протестировать прогнозирующие алгоритмы. Эмпирическим путем установлено, что период чисел Вольфа равен примерно 11 годам. Результат тестирования алгоритма на числах Вольфа представлен на рисунке 4.11. Как видно из графика – первый спрогнозированный период достаточно точен, однако далее уже происходит расхождение.

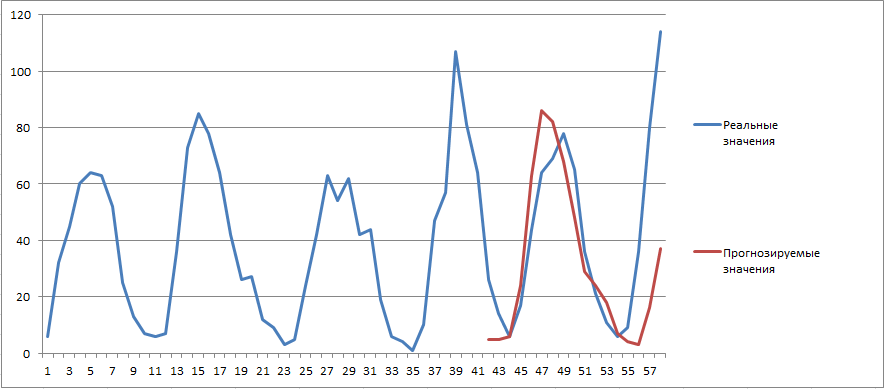


Рисунок .– сравнение прогноза с реальными значениями

### 4.3 Распределение грузов для доставки

Основной целью проектировки любой логистической системы является оптимизация путей доставки грузов от точки отправителя до точки получателя. В общем случае, чтобы распланировать решение этой проблемы, необходимо решить логистическую задачу. Для этой задачи существуют готовые известные решения, такие, как например Open Door. Они предоставляют огромный функционал для решения транспортных логистических задач по оптимизации путей доставки грузов. Однако большинство из этих программ ориентировано на доставку наземным транспортом. Также большинство из готовых решений ориентировано на компании производители. Это означает, что у компании есть собственные средства передвижения, с помощью которых эти компании доставляют свои грузы потребителям. Обычно это наземные грузовые автомобили.

При решении такой задачи, географическая карта разбивается на зоны. В каждой зоне находится условный завод. Завод является условным, т.к. это может быть просто склад, а товар на данный склад в большинстве случаев доставляется с помощью сторонней специализированной логистической компании, которая оперирует большими объемами грузоперевозок. Т.е. приблизительно такой компанией, для которой подошла бы разрабатываемая система. За каждую зону отвечает конкретный завод, от которого доставляются грузы только в точки собственной зоны и не доставляются в чужие зоны. Каждый завод оперирует конкретными единицами транспорта. Весь транспорт обязательно возвращается на завод.

В случае логистической компании, которая ориентирована на доставку грузов не “от заказчика до порога” а “от пункта приема до пункта выдачи”, такие возможности устраивают не полностью. В рассматриваемом случае обязательно необходимо учитывать то, что груз может и, скорее всего, будет иметь конечную точку на очень больших расстояниях. Вплоть до межконтинентальных перевозок. Очевидно, что в таком случае необходимо рассматривать возможность перегрузки грузов с одного транспорта на другой и обратно, возможность выгружать некоторые грузы в определенных пунктах приема и выдачи так, что потом этот груз будет подобран другим транспортным средством, следующим по другому, более оптимальному маршруту.

Еще одним отличием решения рассматриваемой задачи от существующих решений является то, что у существующих решений в составлении маршрутов движения ключевую роль играет конечный пункт груза, т.е. то, куда его необходимо доставить, а транспорт подстраивает маршрут своего движения под доставляемые грузы. В нашей же задаче, определяющую роль играет маршрут движения транспорта. Изначально планируется некоторое путешествие определенного корабля, составляется его расписание и маршрут следования, и только после этого к текущему кораблю привязываются некоторые грузы. Т.е. в данных задачах, идет смещение приоритетов с грузов на транспорт. Так же в существующих открытых решениях, в отличие от разработанной системы, не происходит анализ того, как до этого производилась доставка грузов, а соответственно не проектируется возможность задержек на некоторых участках. Данная возможность помогает отследить и сравнить, как конкретная задержка повлияет на картину распределения грузов в общем представлении, и насколько изменит поток доставки в глобальном смысле. По всем этим причинам, крупные логистические компании обычно производят разработку собственного программного обеспечения, которое будет учитывать все их пожелания, всю специфику их деятельности, удовлетворять необходимым требованиям по качеству и устойчивости. Соответственно самые мощные инструменты для решения таких задач являются закрытыми и не распространяются в свободном доступе. Кроме того, многие разработчики открытого программного обеспечения накладывают ограничение на коммерческое использование своего продукта.

Таким образом, было решено реализовать алгоритм, который бы по заданным кораблям, портам и маршрутам кораблей между этими портами строил распределение заданных грузов по кораблям. Было проанализировано множество вариантов, каким способом было бы лучше воспользоваться. Существует множество детерминированных алгоритмов для решения транспортной задачи. Однако появляется сложность в том, что транспортная задача взаимодействует с однородными грузами, что в свою очередь нам не подходит. Иначе контейнеры могут быть перепутаны и соответственно будут доставлены не в корректные места. Так же возникает сложность с тем, что транспорт ходит строго по расписанию, т.е. какую-то часть времени может ходить часто, какую-то вообще не ходить, а какую-то – ходить, но с задержками. И классические подходы к решению транспортных задач используют однородные средства передвижения. Таким образом, применение таких решений как линейное программирование в данной задаче приносит много сложностей.

За последние годы в прикладных задачах значительно вырос интерес к возможности применения генетических алгоритмов. Применение генетических алгоритмов может помочь найти решение во многих областях, благодаря определенной универсальности. Генетические алгоритмы жестко описывают правила и операции, которых необходимо придерживаться для выявления наилучшего решения, а так же формат представления рассматриваемых данных. Т.е. получается некоторый шаблон последовательности действий и применяемых операций, а сами операции уже реализуются в соответствии со спецификацией исходных данных, зависимостями между этими данными и конечной целью.

Существуют различные формы, в которых генетические алгоритмы применяются к решению научных задач. Например, они используются при создании структур, таких как автоматы или сети сортировки; находят отражение в проектировании нейронных сетей, используются при создании поведения и искусственного интеллекта для различных ЭВМ; и, естественно, применяются по своему прямому назначению – моделирование поведения и развития эволюций в биологических областях.

Однако применять генетические алгоритмы можно и в другой сфере – оптимизация многопараметрических функций. Большое количество реальных прикладных задач может быть сведено к поиску оптимального значения рассматриваемой функции. Иногда необходимо определить значения параметров функции, на которых достигается оптимальное значение функции. Иногда, глобальный максимум (минимум) не нужен, а для ответа достаточно найти любой локальный максимум или просто значение функции большее заранее заданной константы. В любом случае применение генетических алгоритмов возможно. Эта технология оказывается очень гибкой в том смысле, что ее можно применить практически к любой задаче поиска. При чем, не имеет значения, какие сущности будут рассматриваться как множество, в котором будет производиться поиск. Основное требование, чтобы эти сущности были описаны в такой форме, к которой можно было бы применить стандартные операции генетических алгоритмов. Мощность этого подхода состоит в его способности к оптимизации многопараметрических параметров, которая используется в сотнях прикладных программ, от проектирования самолетов, до настраивания параметров алгоритмов для поиска стационарных состояний дифференциальных уравнений.

Генетические алгоритмы являются эффективным алгоритмом поиска, который конкурирует с другими алгоритмами. Эффективность применения генетических алгоритмов зависит от реализации самого алгоритма, от того как будут закодированы решения, и многих параметров. Существует так же много критики в адрес генетических алгоритмов. В частности, в самом начале применения практики генетических алгоритмов, утверждалось, что слабым местом в работе генетических алгоритмов является реализация фитнесс функции. В некоторых задачах для того чтобы вычислить фитнесс функцию требовалось значительное количество вычислительных мощностей и времени. Однако, с процессом невероятного быстрого роста производительности ЭМВ, применение генетических алгоритмов все же стало возможным. Так же, существует критика в адрес того, что в генетическом алгоритме производится повторное вычисление фитнесс функции, что не является эффективным. Однако если из-за этого теряется слишком много времени – можно применить различные подходы, которые смогут оптимизировать этот недостаток, например, такие как ленивые вычисления, когда значение будет вычисляться только один раз по мере необходимости, а после запоминаться. При этом если ген закодировать некоторым образом и получить значение, подобное значению хэш-функции, то для одинаковых генов значение так же не придется вычислять повторно. Безусловно, вся критика в адрес генетических алгоритмов и возможности оптимизации и преодоления слабых мест только порождает больше интереса к исследованию возможности применения генетических алгоритмов.

### 4.4 Реализация алгоритма распределения грузов

Рассматриваемую задачу распределения грузов по кораблям можно отнести к задаче поиска. Для этого необходимо построить граф, вершины которого – порта, т.е. пункты приема и выдачи грузов. Соединять эти вершины будут ребра, которые характеризуют наличие корабля, в маршруте которого эти порты являются соседними. При этом граф является ориентированным. Таким образом, если существует ребро, направленное из порта *А* в порт *Б*, то следовательно существует корабль, который плывет из порта *А* в порт *Б*. Мы будем рассматривать множество всевозможных распределений грузов по кораблям и соответствующую стоимость этого распределения. Тогда задача сводится к тому, чтобы из всего множества найти элемент с минимальной стоимостью. Очевидно, что данная задача NP-полная и могла бы решаться методом полного перебора, однако, при больших данных такое решение может занимать слишком много времени. Таким образом, было принято решение реализовать решение задачи разбиения грузов по кораблям с помощью генетического алгоритма.

Генетические алгоритмы используют три операции: мутация, скрещивание и вычисление фитнесс функции. Генетические алгоритмы основаны на моделировании эволюционного процесса. В каждый момент времени живет некоторая популяция, после чего над выбранными некоторым образом генами производятся операции скрещивания и мутации.

Для реализации генетического алгоритма изначально требуется создать начальную популяцию. Как было описано ранее, для создания начальной популяции не отводится много времени и сил. Достаточно просто создать некоторую популяцию, отвечающую заранее определенным требованиям. Таким образом, для создания начальной популяции рассматривались поочередно все грузы, которые необходимо доставить. В контексте графа, описанного выше, для каждого груза есть начальная точка – порт отправления и конечная точка – порт прибытия. Для каждого груза на рассматриваемом графе строится путь из начальной точки в конечную точку случайным образом. Таким образом, построенный для каждого графа путь не является оптимальным, но это и отвечает нашим требованиям. Поскольку, если для каждого груза строить некоторый оптимальный путь, то мы стремились бы к оптимальному решению для каждого груза в отдельности, т.е. если бы мы перевозили только этот конкретный груз, а остальных бы просто не существовало. Очевидно, что в нашем случае необходимо рассматривать все грузы в совокупности и для оптимальности всей системы, многие грузы могут пойти не по оптимальному для себя и не всегда самому очевидному пути.

Далее производилась реализация операции скрещивания. Идея для реализации операции скрещивания была взята из задачи коммивояжера. В задаче коммивояжера ген – последовательность городов. Для скрещивания берутся два гена, в них выбирается некоторая точка – точка разреза. И гены обмениваются своими половинами. Т.е. левая половина первого гена соединяется с правой половиной второго гена, а левая половина второго гена – с правой половиной первого гена. Далее производятся операции по поддержанию баланса между посещаемыми городами, т.к. один и тот же город может присутствовать в левой и в правой части. В таком случае в одной из частей повторяющиеся города заменяются на те, которых не хватает. Для нашей же задачи данный алгоритм не подходит и должен быть расширен. Т.к. в задаче коммивояжера рассматривается полный граф и поэтому возможно подобное оперирование с вершинами графа, в нашей же задаче в общем случае граф не является полным и такое решение не подходит. Однако данная идея может быть взята за основу и усовершенствована. Было принято решение ввести в рассмотрение точки пересечений для двух путей. Точки пересечений для двух путей – такие точки в графе, которые присутствуют в обоих путях. Очевидно, что для одного и того же груза, если рассматривает его в двух разных популяциях – точек пересечений будет как минимум две – это начальная точка и конечная точки. Это свойство говорит о том, что всегда будет возможность произвести операцию скрещивания в данном контексте. Таким образом, мы получаем множество точек, в которых два пути пересекаются. Из этого множества мы случайным образом выбираем одну точку – точку разрыва. В новый, дочерний путь одну половину мы берем из первого пути, а вторую половину – из второго пути. Таким образом, мы смогли обойти несоответствие по городам в разных половинах и избежали создания путей по несуществующим дугам. При этом дочерний элемент в равной степени наследует свойства обоих родителей. Таким образом, реализованная операция скрещивания генов удовлетворяет всем требованиям, которые предъявляют генетические алгоритмы. Пример скрещивания двух путей, когда точка пересечения находится посредине, приведен на рисунке 4.12.

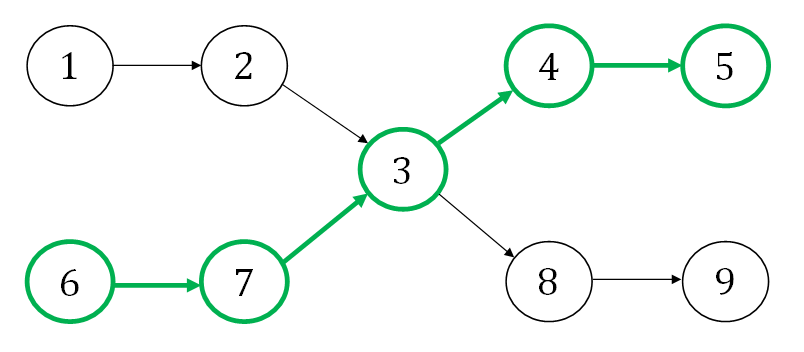


Рисунок . – пример скрещивания двух путей

Так же в данной работе для генетического алгоритма, была реализована операция мутации. Реализованная операция мутации оперирует, прежде всего, с путем который проходит груз. Мутация в данной системе производится следующим образом. Изначально происходит оценка всех вершин. Отбираются вершины только со степенями больше двух. Если начальная или же конечная вершины не отобраны – оценивается их степень. Начальная и конечная вершины могут быть добавлены к уже отобранным вершинам, если их степень больше единицы. Далее из всех отобранных вершин выбираются две случайным образом. После чего между этими вершинами строится путь случайным образом. При построении учитывалось направление движения корабля. Для мутации гены выбирались случайным образом. В каждом выбранном гене случайное число путей для грузов мутировало в новые пути. Операция мутации в данной реализации имеет много схожестей с операцией инициализации. Кроме того она предназначается для вывода множества решений из локальных минимумов.

В данной работе была произведена реализации вычисления функции приспособленности. Значение фитнес функции эквивалентно значению стоимости всех перевозок при соответствующем распределении грузов. В данной реализации принято было вычислять стоимость как произведение перевозимой продукции и время, отводимое на перевозку. В аналогичных задачах, часто в качестве второго параметра указывают насколько далеко необходимо перевезти груз, т.е. расстояние для перевозки. Это было бы оправдано, если бы мы пускали корабли в соответствии с тем, куда надо доставить груз. Однако в данной задаче груз привязывается к кораблю с заданным маршрутом. Поэтому, целесообразно вторым параметром рассматривать время. Очевидно, что время оплачивается, и в общем случае оплата прямо пропорционально зависит от времени. Поэтому в такой интерпретации мы временем характеризуем оплату и данной оплате можно придать множество прикладных значений, например: плата за нахождение груза на корабле, плата за обслуживания груза, плата за обслуживание контейнеров, плата за пользование контейнерами и т.д. Таким образом, приспособленность в общем виде вычисляется по формуле 4.1.

|  |  |
| --- | --- |
|  | (4.1) |

Таким образом, ставится задача о вычислении времени, необходимом для перевозки всех грузов. На вход фитнесс функции подается план движения всех грузов. Т.е. для каждого груза есть начальная точка, конечная точка и путь, соединяющий начальную и конечную точки. Основная задача фитнесс функции смоделировать процесс движения кораблей и вычислить стоимость за все перевозки. Данная задача имеет некоторую схожесть с задачей поиска в ширину. Решение было решено реализовать с помощью приоритетной очереди. Элементом очереди является структура, содержащая груз, его текущую вершину и время когда он был доставлен в эту вершину. При этом приоритет элементов определяется по времени. Таким образом, мы каждый раз достаем элемент с наименьшим временем и рассматриваем его.

Последовательность обработки каждого элемента заключается в следующих пунктах:

1. Определяем текущий порт этого элемента и следующий порт, в который он должен прибыть.
2. Находим корабль, на котором рассматриваемый груз сможет отправиться в следующий порт. Искомый корабль можно определить по двум факторам. Первый – свободного места оставшегося на корабле должно хватать, чтобы увезти данный груз. Второй – корабль должен отправляться из порта тогда, когда данный груз уже будет в этом порту. Если такого корабля мы не можем найти – план не подходит, заканчиваем вычисление.
3. Достаем из приоритетной очереди все элементы, которые мы можем увезти на этом корабле. Искомые элементы можно определить по двум факторам. Первый – вес груза меньше, чем оставшееся место на корабле. Второй – элемент прибывает в данный порт не позже, чем корабль должен отправиться. Таким образом, мы получаем множество всех элементов, претендующих на данный корабль.
4. Заполняем корабль грузами насколько можем.
5. Перевозим грузы. В данном смысле перевозка обозначает изменение текущей вершины и времени.
6. Модифицируем общую стоимость с помощью соответствующей формулы.
7. Добавляем в очередь все грузы, кроме тех, которые достигли своей конечной цели.
8. Если очередь не пуста, переходим к шагу 1.

В процессе вычисления мы заполняем корабль выбранными грузами на шаге 3. В общем случае нам необходимо выбрать, какие грузы нам выгоднее перевозить на данном корабле. Следуя выбранной формуле вычисления приспособленности, как произведения объема грузов и времени, нам важно минимизировать именно время. Потери во времени в нашем случае возникают тогда, когда груз ждет своего корабля в порту. При этом не все грузы могут быть перевезены с помощью корабля, т.к. может не хватить места. Может возникнуть такая ситуация, что груз не будет перевезен на корабле, но на последующем его все же перевезут. При чем, далее встретится такая вершина, на которой груз будет ждать своего корабля большее время, чем, если бы груз был перевезен сразу на текущем корабле. Стоит отметить, что в реальной жизни такое опоздание случается часто. Но, если мы рассматриваем популярный маршрут, корабли по которому ходят постоянно, то потерь данная проблема практически не вызывает. Считается, что выгоднее сначала перевозить грузы в проблемные области, где корабли ходят реже, для обеспечения большей заполняемости. Очевидно, что при таком подходе, влияние задержек снижается. Т.к. чем раньше мы будем планировать привезти груз в проблемный участок – тем больше мы можем опоздать на практике. Для того чтобы выбрать какие грузы выгоднее перевозить в данный момент – необходимо проследить весь путь следования грузов. Рассмотрим выбор более приоритетного груза из двух – “А” и “Б”. Для данного выбора будем рассматривать их влияние на общую стоимость доставки вех грузов и выберем наиболее оптимальный вариант. Рассмотрим два случая. Первый случай, когда груз “А” перевозится на текущем корабле, груз “Б” – на следующем. Второй случай, когда груз “Б” перевозится на текущем корабле, груз “А” – на следующем. Подсчитаем стоимость таких перевозок для каждого случая от текущей вершины, до конечной вершины груза “А” и конечной вершины груза “Б”. Приоритетным будет вариант с меньшей стоимостью. По данному принципу необходимо отсортировать все грузы. Таким образом, мы будем доставлять грузы как можно раньше на проблемные участки всей сети, т.е. те участки, где корабли ходят редко и для них необходимо заполнение как можно большим количеством грузов.

Для однозначности покажем, что такой оператор сравнения является транзитивным. Иначе результат сортировки не будет однозначным. Рассмотрим три элемента “A”, “B” и “C” с соответствующими объемами “a”, “b” и “c”. Пусть “A” < “B” и “B” < “C”. Тогда “A” < “B” выражается формулой 4.2.

|  |  |
| --- | --- |
|  | (4.2) |

В формуле 4.2 слева отражена ситуация, когда сначала перевозят груз “A” , а на следующем корабле – груз “B”. При этом - время отправления из текущего порта. - время прибытия в следующий порт первого корабля. - время прибытия в следующий порт второго корабля. ,,, – характеризуют времена прибытия в конечный порт, каждое значение для своего коэффициента. Справа отражена обратная ситуация. Тогда аналогично можно представить выражение “B” < “C” с помощью сравнения 4.3.

|  |  |
| --- | --- |
|  | (4.3) |

Временные коэффициенты ,- характеризуют времена прибытия в конечный порт груза “C”, для случая, когда он отправляется на втором и на первом корабле соответственно. Сократим слагаемые в неравенствах 4.2 и 4.3 и в неравенстве 4.3 перенесем слагаемые с коэффициентом в противоположные части. После данных преобразований получим неравенства в виде 4.4.

|  |  |
| --- | --- |
|  | (4.4) |

Неравенства 4.4 мы можем сложить. А к полученной сумме прибавим и отнимем коэффициенты и слева и и справа. Результат сложения и результат суммы с коэффициентами представлены неравенствами 4.5.

|  |  |
| --- | --- |
|  | (4.5) |

Второе неравенство в формуле 4.5 характеризует отношение элементов “A” < “B”. Таким образом, доказана транзитивность оператора отношения на множестве элементов. Следовательно, мы можем отсортировать грузы по приоритету, и при этом результат этой сортировки будет однозначным.

Произведем тестирование работы данного алгоритма. Изначально, для простоты понимания и оценки работы возьмем 9 портов и пустим по ним корабли восьмеркой, так, чтобы обеспечить альтернативные пути доставки грузов. Расположение портов представлено на рисунке 4.13.

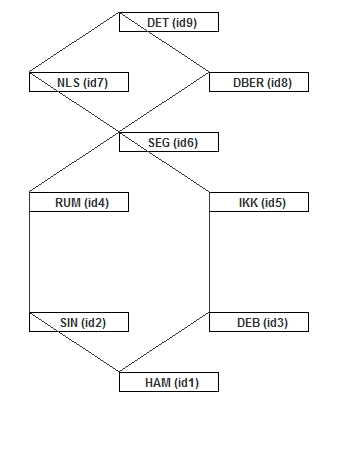


Рисунок . – расположение портов

Корабли пустим в обе стороны по верхней части, т.е. по кругу в портах с идентификационными номерами 9, 7, 6, 8. И по нижней части – в портах с идентификационными номерами 3, 5, 6, 4, 2, 1 – так же в обе стороны. Возьмем 20 грузов и поместим в порт №1, будем доставлять их в верхнюю часть – в порт №9. До верхней части возникают два пути. Первый – последовательность 1, 2, 4, 6. Второй – 1, 3, 5, 6. В верхней части, от порта №6 так же два пути – 6, 7, 9 и 6, 8, 9. Изначально протестируем алгоритм при равном времени по обоим портам. Результат представлен на рисунке 4.14.

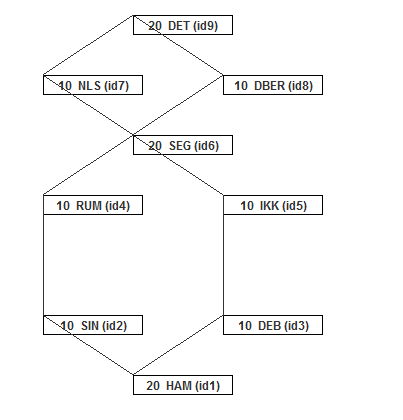


Рисунок . – распределение при равных временах плавания

Как видно из рисунка – грузы распределились равномерно по обоим возможным маршрутам. Теперь попробуем ввести задержку на последовательности между 1, 2, 4 и 6. При относительно малых задержках в несколько часов – распределение не меняется, однако, если мы водим задержку в два и более десятка часов – поток грузов заметно перераспределяется. Результат работы алгоритма при задержке в 14 часов между портами 1 и 2 и 18 часов между портами 2 и 4 представлен на рисунке 4.15. Как показывает практика и как приводилось в предыдущем параграфе – задержки в 20 – 50 часов не являются редкостью. На рисунке 4.15 видно, что существует некоторое пороговое значение задержек. После прохождения этого значения грузы начинают перераспределяться по другим путям во избежание возникновения больших затрат. По левой ветви отправлено 8 грузов, а по правой – 12. Если поставить задержку на левой ветви еще больше – получим распределение, представленное на рисунке

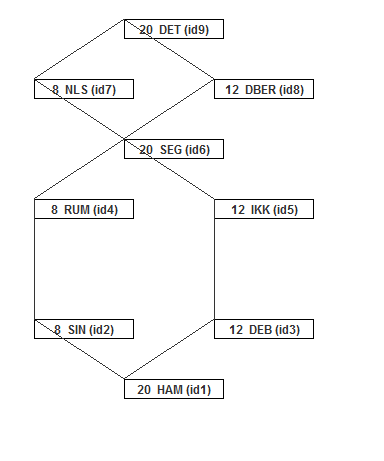


Рисунок . – распределение грузов для небольших задержек между 1 и 2, а так же 2 и 4 портами.

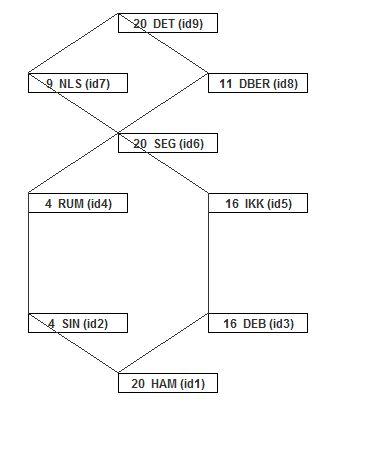


Рисунок . – распределение грузов при дальнейшем увеличении задержек на левой ветви

Таким образом, мы получаем перераспределение и значительное отклонение от стандартного плана при задержках. Здесь стандартный план – равное количество грузов с левой и с правой стороны. Проведем сравнение стандартного плана и плана, выдаваемого алгоритмом при возникновении задержек. На рисунке 4.17 приведены графики значений фитнес функции в зависимости от величины задержки на левой ветви. В таблице 4.3 указан выигрыш в процентном соотношении.

Таблица 4.3 – сравнение графиков фитнесс функции для стандартного плана и плана, выдаваемого генетическим алгоритмом

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Задержка | 0 | 10 | 20 | 30 | 40 | 50 | 60 | 70 |
| Стандартный план | 65600 | 65600 | 68800 | 68800 | 75200 | 75200 | 78400 | 78400 |
| Генетический алгоритм | 65600 | 67200 | 72000 | 72000 | 80000 | 81600 | 86400 | 86400 |
| Выигрыш, % | 0 | 2.38 | 4.4 | 4.4 | 6 | 7.84 | 9.25 | 9.25 |
| Задержка | 80 | 90 | 100 | 110 | 120 | 130 | 140 | 150 |
| Стандартный план | 84800 | 84800 | 84800 | 84800 | 84800 | 88000 | 88000 | 88000 |
| Генетический алгоритм | 94400 | 96000 | 100800 | 100800 | 108800 | 108800 | 110400 | 115200 |
| Выигрыш, % | 10.1 | 11.6 | 15.8 | 15.8 | 19.11 | 19.11 | 20.28 | 23.6 |

Как мы видим из рисунка 4.17 и таблицы 4.3, если в сети существует альтернативный маршрут – данный генетический алгоритм значительно улучшает стоимость доставки и содержания грузов. Из таблицы видно, что на средних задержках выигрыш затрат достигает 5 – 10 процентов. Дальше с увеличением задержки до высоких значений выигрыш доходит до 20 процентов. Полученный результат, несомненно, важен при оперировании большими объемами грузов.

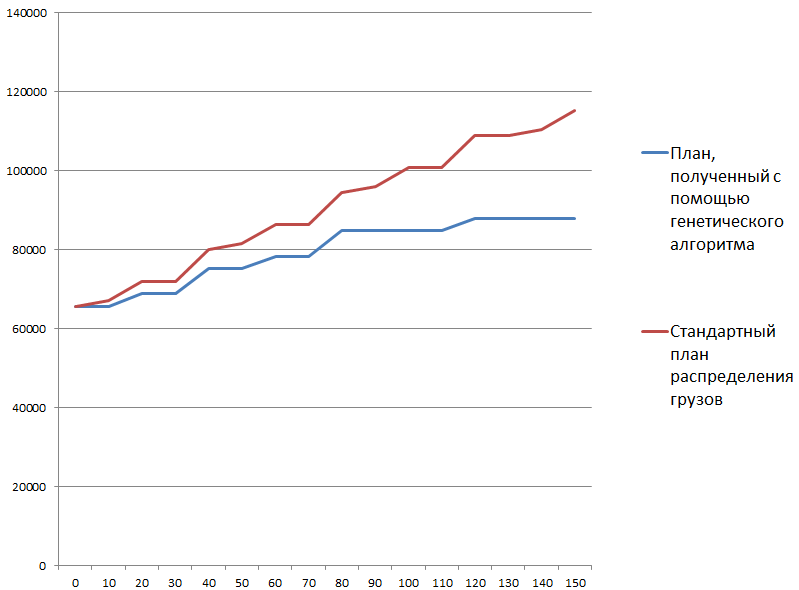


Рисунок . – сравнение графиков фитнесс функции для стандартного плана и плана, выдаваемого генетическим алгоритмом

Время выполнения данного алгоритма заняло 1321 миллисекунду на данных представленных выше. Так же было произведено тестирование алгоритма на случайных данных, но реальных размерностей. При 1000 грузах, 100 портах и 20 суднах время выполнения заняло 1236165 миллисекунд, т.е. примерно 20 минут. Размерность этих данных сопоставима с месячным грузооборотом отдельно взятой компании, обслуживающей российские воды. При дальнейшем увеличении количества грузов до 10000, время работы программы составило 3611845 миллисекунд. Т.к. планирование грузоперевозок производится заранее, а не в режиме реального времени, то применение данного алгоритма возможно. Кроме того, при условии уже спрогнозированных задержек, применение данного алгоритма значительно помогает оценить масштабы влияния задержек на уже спроектированный план распределения грузов. И при сильном отличии от плана, выдаваемого алгоритмом – внести соответствующие корректировки. Например – увеличить скорость кораблей, отменить рейсы или изменить насколько возможно маршрут.

## ЗАКЛЮЧЕНИЕ

В данной работе:

* изучены гибкие методы разработки приложений в мульти-платформенной системе ГИС;
* рассмотрен метод прогнозирования SSA;
* разработана модель данных для транспортной логистической системы в базе данных DB2;
* реализовано приложение для отображения кораблей, портов и терминалов на карте;
* реализовано приложение, отображающее перегружаемые грузы;
* реализовано приложение репорт для оперативной оценки задержек кораблей;
* произведена реализация алгоритма SSA;
* выполнено тестирование алгоритма SSA для задержек кораблей и для чисел Вольфа;
* рассмотрена возможность применения генетических алгоритмов;
* реализованы операции скрещивания и мутации с учетом специфики исследуемой сети;
* реализована функция приспособленности, с учетом специфики исследуемой сети с приоритетной доставкой грузов в зоны, где корабли ходят реже;
* реализован генетический алгоритм, распределяющий заданные грузы, по транспортным судам;
* произведено тестирование алгоритма на входных данных с альтернативным маршрутом и возникающими задержками;
* произведено исследование скорости работы генетического алгоритма на различных наборах входных данных;
* оценена получаемая с помощью генетического алгоритма выгода в случае возникновения задержек.

Таким образом, разработанная система предоставляет возможность отображения различных данных о кораблях, портах, грузах и перегрузках грузов. Так же предоставлена возможность отображения истории и прогнозирования задержек кораблей. С помощью этой функциональности данное решение дало возможность уменьшить непредсказуемость возникновения задержек. Так же данное решение дало возможность оценить масштабы влияния возникающих задержек на финансовые потери. Соответственно, представилась возможность выявления проблемных областей в сети, в которых величина задержки превосходит пороговую, из-за чего стоимость перевозок на участке начинает значительно возрастать. Таким образом, получилось комплексное решение по оптимизации путей доставки грузов с помощью морского транспорта. Разработанный программный модуль внедрен в процесс управления грузовыми перевозками морского транспортно-логистического предприятия.

## СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Сухов, С.А. Основы программирования на Java: учебное пособие / С.А. Сухов – Ульяновск: УлГТУ, 2006 – 88 с.
2. Фрэнсис, Т. АЙ БИ ЭМ профессиональный WS сервер / Т. Фрэнсис. – Нью-Йорк: Джон Войли и Сонс, 2004. – 792 с.
3. Райнс, К. Разработка и развертывание Java приложений / К. Райнс. – Нью-Йорк: АЙ БИ ЭМ Рэдбукс, 2013 – 406 с.
4. Двигатель SQL запросов в DB2 для i5/OS / Х. Бедойя [и др.]; под общ. ред. Х. Бедойя – Нью-Йорк: АЙ БИ ЭМ Рэдбукс, 2006 – 206 с.
5. Радченко, М.Г. 1С: Предприятие 8.2. Версия для обучения программированию / М.Г. Радченко, Е.Ю. Хрусталева – Москва: 1С-Паблишинг, 2011 – 2228 с.
6. Шурыгин, А.М. Математические модели прогнозирования / А.М. Шурыгин – Москва: Горячая Линия – Телеком, 2009 – 180 с.
7. Лоскутов, А. Ю. Анализ временных рядов / А. Ю. Лоскутов. – Физфак МГУ, 2001. – 106 с.
8. Курейчик, В.М. Генетические алгоритмы и их применение / В.М. Курейчик – Таганрог: Изд-во ТРТУ, 2002 – 244 с.