Добрый день, уважаемые члены аттестационной комиссии. Тема моей ВКР – **«Разработка системы автоматического развертывания микросервисных приложений в облачной инфраструктуре на основе алгоритма комбинаторной оптимизации».**

Слайд 2. Почему начал исследование. Актуальность. Новизна

Данная работа посвящена рассмотрению способа минимизации кол-ва занимаемых физических и виртуальных серверов на предприятии «Северсталь-инфоком» на примере внедряемого проекта систем адресного учета продукции.

Опишем проблему, используя пирамиду автоматизации, представленную на слайде 2. Можно выделить 2 уровня систем – уровень автоматизированных систем управления предприятием и уровень автоматизированных систем управления технологическим процессом. И если на верхнем уровне применяются такие средства управления ресурсами, как контейнеризация и оркестрация, то на АСУ ТП такие инструменты не применяются из тех соображений, что довольно сложно спрогнозировать надежность данных систем.

Исследование на данную тему является актуальным, т.к. на данный момент нет программных решений управления использованием ресурсов серверов на уровне АСУ ТП. Предприятие заинтересовано в данном исследовании, т.к. стремится максимально снизить издержки на покупку новых серверов.

Соответственно, требуется собственное решение.

Слайд 3.

Цель: минимизация количества задействованных виртуальных и физических серверов при распределении программных компонентов на серверной инфраструктуре.

Объект: инфраструктура развертывания программного обеспечения на физических и виртуальных серверах центра обработки данных АО «Северсталь-инфоком».

Предмет: методы оптимизации автоматического развертывания микросервисных приложений на серверах.

Слайд 4.

1. Проанализировать существующие решения по развертыванию ПО на серверных фермах. Провести патентный обзор.
2. Рассмотреть существующие алгоритмы оптимизации, разработать математическую модель системы, адаптировать алгоритмы и реализовать симулятор системы для выбора наиболее подходящего алгоритма оптимизации.
3. Реализовать автоматического развертывания ПО, провести экспериментальные исследования работы системы.
4. Выполнить технико-экономическое обоснование проекта.

Слайд 5.

Научная новизна работы заключается в применении генетического алгоритма при решении задачи оптимального распределения программного обеспечения на серверах и разработке функций этого алгоритма:

* скрещивания;
* мутации;
* функции приспособленности.

Слайд 6.

На сегодняшний день микросервисная архитектура становится все более популярной в серверных приложениях, в связи с чем все больше применяется контейнеризация – это легковесная виртуализация и изоляция ресурсов. Самым распространенным инструментом автоматизации развертывания контейнеризованных приложений является Docker, а средством масштабирования и координации в условиях кластера – Kubernetes.

Патентный обзор показал, что существуют решения для управления ресурсами сервера, но все они сводятся к использованию либо статических правил распределения ресурсов, либо не ставят цель минимизировать кол-во занятых серверов.

Слайд 8.

На слайде 8 представлена схема, на которой показаны инфраструктура до и после мероприятий по минимизации занимаемого пространства.

Слайд 9.

Задача минимизации количества используемых серверов в терминах комбинаторной оптимизации формулируется следующим образом: «имеется коллекция программных компонентов (далее будем называть их сервисами) и коллекция виртуальных машин (далее будем называть их серверами). Необходимо распределить все сервисы на минимальное количество серверов предполагая, что один сервис может располагаться только на одном сервере».

Согласно мат модели необходимо, чтобы кол-во серверов было наименьшим возможным, при условии соблюдения следующих ограничений

1. Размер файла не должен превышать объем ПЗУ сервера
2. Требуемое кол-во ОЗУ не должно превышать ОЗУ сервера
3. Тип операционной системы сервиса и сервера должны совпадать

Мат модель:

где еще вставить ограничение на тип ОС?

Критерием оптимальности в данном случае будем считать число серверов . Решение считается тем более оптимальным, чем ниже значение .

Вышеописанная задача сводится к задаче об упаковке в контейнеры, которая, в свою очередь, является NP-полной задачей. Трудоёмкость таких задач экспоненциально растёт с увеличением объема данных. Для NP-полных вычислительных задач не существует алгоритма решения, способного вернуть результат за полиномиальное время, поэтому рассмотрим ряд эвристических алгоритмов, позволяющих найти решение, находящееся близко к оптимальному. В качестве таких алгоритмов были выбраны 3 категории:

1. Простейшие алгоритмы NF, FF, BF, FFD, BFD. Они похожи, первые 3 это online алгоритмы, последние 2 нет. Был выбран последний, как алгоритм с полным доступом ко всем контейнерам и наиболее подходящий по смыслу к задаче.
2. Алгоритм имитации отжига. Алгоритм оптимизации, использующий упорядоченный случайный поиск на основе аналогии с процессом образования в веществе кристаллической структуры с минимальной энергией при охлаждении.
3. Генетический алгоритм. Цель генетического алгоритма при решении задачи оптимизации состоит в том, чтобы найти приближённое решение, близкое, но не гарантированно оптимальное решение.

Для того, чтобы выбрать наиболее подходящий для решения поставленной задачи алгоритм, был реализован симулятор инфраструктурной среды, состоящий из сервиса, возвращающего среду для оптимизации и сервиса, принимающего среду на вход и оптимизирующего ее с применением вышеперечисленных алгоритмов. Реализация алгоритма имитации отжига и генетического алгоритма включала в себя подбор необходимых параметров и разработку функции энергии и приспособленности, соответственно.

Слайд 11.

На слайде представлена функция приспособленности генетического алгоритма, а также, другие параметры, используемые в его работе.

Слайд 12. На слайде представлена схема многоточечного скрещивания. Согласно ей, сервис с четным индексом располагается в хромосоме-потомке на том же сервере, на котором он расположен на первой родительской хромосоме. Сервис с нечетным индексом располагается на хромосоме-потомке на том же сервере, на котором он расположен на второй родительской хромосоме.

Слайд 13. Рассмотрим результаты работы алгоритмов.

Сравнение производилось по выбранным экспертным путем критериям:

1. Время работы алгоритма
2. Вероятность ошибочного решения
3. Заполняемость серверов

Получены следующие усредненные данные по результатам 50 запусков на тестовой выборке.

По данным результатам можно видеть, что время работы алгоритма находится в пределах установленного (установлено 10 минут), вероятность ошибочного решения у генетического алгоритма самая низкая, а миниамльное число использованных серверов также у генетического.

Заполненность сервисов также демонстириуется на графиках, где бордовая линия – результаты заполненности серверов при решении задачи генетическим алгоритмом.

Время работы для различных алгоритмов.

|  |  |
| --- | --- |
| Название алгоритма | Время работы алгоритма, с |
| Алгоритм BFD | 0,0022 |
| Генетический алгоритм | 1,2631 |
| Алгоритм имитации отжига | 0,2365 |

Таблица 2.10. Зависимость количества свободных серверов от количества размещаемых сервисов

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | % свободных серверов | | | | | | | | | | |
| Кол-во сервисов | 0 | 5 | 10 | 15 | 20 | 25 | 30 | 35 | 40 | 45 | 50 |
| Алгоритм BFD | 100 | 60 | 40 | 20 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Генетический алгоритм | 100 | 60 | 60 | 60 | 60 | 60 | 40 | 40 | 0 | 0 | 0 |
| Алгоритм имитации отжига | 100 | 60 | 60 | 60 | 60 | 40 | 40 | 40 | 0 | 0 | 0 |

Вероятность ошибочного решения для различных алгоритмов.

|  |  |
| --- | --- |
| Название алгоритма | Вероятность ошибочного решения |
| Алгоритм BFD | 0 |
| Генетический алгоритм | 0,01 |
| Алгоритм имитации отжига | 0,07 |

По показателям затраченного времени работа всех алгоритмов укладывается в установленные рамки. Вероятность ошибочного решения у генетического алгоритма имеет минимальное значение, а дисперсия значений заполненности, наоборот (или лучше написать просто про тенденцию к сохранению минимального числа заполненных серверов?), имеет максимальные значения, что говорит о том, что часть серверов заполнена максимально, а оставшаяся минимально возможно. Таким образом, сделан вывод, что наиболее подходящим для решения задачи распределения сервисов по виртуальным машинам (серверам) является генетический алгоритм.

Слайд 18. Разработал ПО

На видео: Для развертывания приложений следует:

1. Создаем компоненты в модуле компонентов
2. В модуле хостов создаем целевую машину
3. Создаем среду развертывания, где указываем какие компоненты развернуть на каких серверах. Добавляем все сервисы только на первый сервер.
4. В модуле оптимизации сервисы распределены на сервера.
5. Разворачиваем данные компоненты на нашей инфраструктуре

На основе выбранного алгоритма реализовано клиент-серверное ПО, позволяющее в автоматическом режиме разворачивать программные компоненты на серверах и оптимизировать их загруженность. Использованные технологии: backend: .NET Core + EFCore. БД: postgres.

Frontend: wpf + prizm + catel + adonisUI.

Монолитный шлюз для API: Ocelot.

Брокер сообщений: RabbitMQ

Приложение запущено в контейнерах Docker.

Слайд 19. Экономическое обоснование

Как показали эксперименты по реструктуризации существующей инфраструктуры с применением разработанного программного обеспечения, возможно высвободить ресурсы на существующей инфраструктуре до 13%. Проект внедрения САУП рассчитан на 6 лет, в течение которых закупается по 8 новых серверов в год на сумму 1 128 000 рублей. При оптимизации существующей инфраструктуры мы освобождаем 2 физических сервера в первый год и по одному в последующие. Соответственно, экономия средств в первый год составит 517 100,6 рублей, а в последующие 274 143,2 рубля. Проект оправдывает затраченные на него средства менее, чем за полгода и позволяет сэкономить более 270 тысяч рублей в год.