1. Почему начал исследование. Новизна

На предприятии существуют датацентры. На них стоят сервера. Сервера занимает программное обеспечение. Покупка серверов стоит денег, поэтому предприятие стремится максимально задействовать существующие мощности / снизить количество занимаемых приложениями серверов. Используя такую модель, как пирамида автоматизации, можно выделить 2 уровня - автоматизированных систем управления технологическим процессом и автоматизированных систем управления предприятием. И если на уровне АСУП применяются такие средства управления ресурсами, как контейнеризация и оркестрация, то на АСУ ТП такие инструменты не применяются. Почему нельзя использовать кубер и докер на L2?

Соответственно, требуется собственное решение.

1. 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».

Требуется найти такое разбиение множества на непересекающиеся подмножества , чтобы сумма размеров сервисов в каждом подмножестве не превосходила заданную вместимость конкретного сервера, и чтобы было наименьшим возможным

Мат модель:

*,*

*,*

где

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.
4. Сделал симулятор, выбрал генетический. Показать сравнение алгоритмов.

Для того, чтобы выбрать алгоритм, наиболее подходящий, был реализован симулятор инфраструктурной среды, состоящий из сервиса, возвращающего среду для оптимизации и сервиса, принимающего среду на вход и оптимизирующего ее с применением этих алгоритмов.

Результаты следующие (50 запусков на тестовой выборке. Результаты усреднены):

Принимая во внимание показатели работы вышеописанных алгоритмов, представляется возможным сделать вывод, что наиболее подходящим для решения задачи распределения сервисов по виртуальным машинам (серверам) является генетический алгоритм, т.к. показатели заполненности серверов при работе данного алгоритма превышают показатели других исследуемых алгоритмов, а вероятность ошибочного решения ниже, чем у алгоритма имитации отжига.

Добавить маленькие графики заполненности CPU, HDD, RAM.

1. Разработал ПО

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.

1. Экономическое обоснование

Как показали эксперименты по реструктуризации существующей инфраструктуры с применением разработанного программного обеспечения, возможно высвободить ресурсы на существующей инфраструктуры 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.

1. Видео работы

Показать видео, как работает ПО.