7. Самые опасные вирусы за всю историю существования компьютеров [Электронный ресурс]. – 2013. – Режим доступа: https://www.osp.ru/pcworld/2014/01/13038813/.

8. Десять самых громких хакерских атак в истории интернета [Электронный ресурс]. – 2014. – Режим доступа: https://republic.ru/posts/l/1139636/.

9. Вирусы, статистика и немного всего [Электронный ресурс]. – 2017. – Режим доступа: https://habr.com/ru/post/357426/.

10. Новая работа для графических процессоров: GPU защитит от вирусных атак [Электронный ресурс]. – 2018. – Режим доступа: https://habr.com/ru/company/1cloud/blog/354526/.

11. «Разрубить Гордиев узел», или преодоление проблем шифрования информации в ОС Windows [Электронный ресурс]. – 2016. – Режим доступа: https://habr.com/ru/company/aladdinrd/blog/304024/.

12. Уязвимости SSD с аппаратным шифрованием позволяют злоумышленникам легко обходить защитные меры [Электронный ресурс]. – 2016. – Режим доступа: https://habr.com/ru/post/428964/.

13. Прозрачное шифрование: преимущества и недостатки [Электронный ресурс]. – 2015. – Режим доступа: https://habr.com/ru/company/cybersafe/blog/251041/3/.

14. Срыв масштабной хакерской атаки на пользователей Windows в России: часть 2 [Электронный ресурс]. – 2018. – Режим доступа: https://habr.com/ru/company/microsoft/blog/351692/.

СРАВНЕНИЕ АЛГОРИТМОВ РЕШЕНИЯ ЗАДАЧИ КОММИВОЯЖЁРА

Астрашаб В. В., Бондарев И. М., Клебанов Д. А.

Белорусский государственный университет информатики и радиоэлектроники
г. Минск, Республика Беларусь

Стройникова Е. Д. — ассистент кафедры информатики

В докладе рассматривается одна из ключевых задач комбинаторной оптимизации — задача коммивояжёра. Формулируется суть задачи, рассматриваются различные алгоритмы её решения, а также анализируются и сравниваются результаты выполнения алгоритмов при различных наборах входных данных.

Задача коммивояжёра — одна из самых известных задач в комбинаторной оптимизации. Она заключается в следующем: «Зная список городов и их координаты, найдите самый короткий путь, проходящий через все города один раз и возвращающийся в исходный город».

Данная задача была впервые сформулирована в 1930 г. и до сих пор является одной из наиболее интенсивно изучаемых задач оптимизации. Ричард Карп в 1972 г. доказал NP-полноту задачи поиска гамильтоновых путей, из чего, благодаря полиномиальной сводимости, вытекала NP-трудность задачи коммивояжёра. На основе этих свойств им было приведено теоретическое обоснование сложности поиска решений задачи на практике.

Задача коммивояжёра используется в логистике, транспорте, проектировании различных коммуникационных сетей и трубопроводов, а также в других сферах.

Вместе с простотой определения и сравнительной простотой нахождения хороших решений задача коммивояжёра отличается тем, что поиск действительно оптимального пути является достаточно сложной задачей.

Существует множество различных способов решения данной задачи, которые можно разделить на несколько групп:

1. Точные алгоритмы — алгоритмы, которые находят гарантированно верное решение.

Примеры:

A. Полный перебор всех возможных путей. Сложность алгоритма $O(N!)$.
B. Алгоритм Хелда — Карпа — алгоритм динамического программирования. Сложность алгоритма $O(N^2 * 2^n)$.
C. Метод ветвей и границ — вариация полного перебора с отсевом подмножества заведомо неоптимальных решений.

Ключевое преимущество алгоритмов данной группы заключается в том, что они гарантированно находят верное решение, чего не позволяют сделать алгоритмы из других групп. Однако на практике эти алгоритмы почти никогда не применяются ввиду огромных временных затрат даже при небольших значениях N.

2. Эвристические алгоритмы — алгоритмы, которые не гарантируют точного решения, однако достаточны для решения поставленной задачи.

Примеры:

A. Деревянный алгоритм — алгоритм решения через построение кратчайшего остовного дерева. Сложность $O(N^2)$.
B. Алгоритм ближайшего соседа — жадный алгоритм, который строит путь, находя самую ближнюю к данной вершину. Сложность $O(N^2)$.
C. 2-opt алгоритм — алгоритм, сводящийся к удалению двух пересекающихся рёбер и вставке новых рёбер, не нарушающих корректности решения. Сложность $O(N^2)$.

К преимуществам данных алгоритмов можно отнести строгую зависимость скорости поиска решения от размера исходных данных.

К недостаткам данных алгоритмов можно отнести низкое качество ответа при увеличении количества городов.

3. Метаэвристические алгоритмы — обобщённые стратегии поиска оптимума в пространстве решений, зависящие от случайности.

Примеры:

A. Муравьиный алгоритм — алгоритм, имитирующий поведение муравьиной колонии, ищущей путь к источнику питания.
B. Имитация отжига — алгоритм, имитирующий физический процесс кристаллизации вещества при понижающейся температуре.
C. Генетический алгоритм — алгоритм, имитирующий эволюционный процесс в природе.
D. Алгоритм клонирующего отбора — разновидность генетического алгоритма, не использующая наследование от нескольких предков.

К преимуществам данных алгоритмов можно отнести простоту реализации, нахождение более оптимальных путей по сравнению с эвристическими алгоритмами.

К недостаткам данных алгоритмов можно отнести зависимость от гиперпараметров, которые подбираются индивидуально для различных наборов исходных данных, а также сложность асимптотического анализа из-за различающихся гиперпараметров.

Для сравнения алгоритмы различных типов были реализованы на языках программирования C++ и Python, а также сгенерированы различные наборы входных данных для тестирования. Полученные результаты были визуализированы с использованием библиотеки Matplotlib.

Анализ результатов тестирования:

1) метаэвристические алгоритмы работают значительно дольше эвристических;

2) пути, полученные эвристическими алгоритмами, длиннее путей, полученных метаэвристическими алгоритмами (рисунок 1);

3) при малых размерах исходных данных высока вероятность нахождения близких к оптимальному решений или же оптимального решения эвристическими и метаэвристическими алгоритмами.

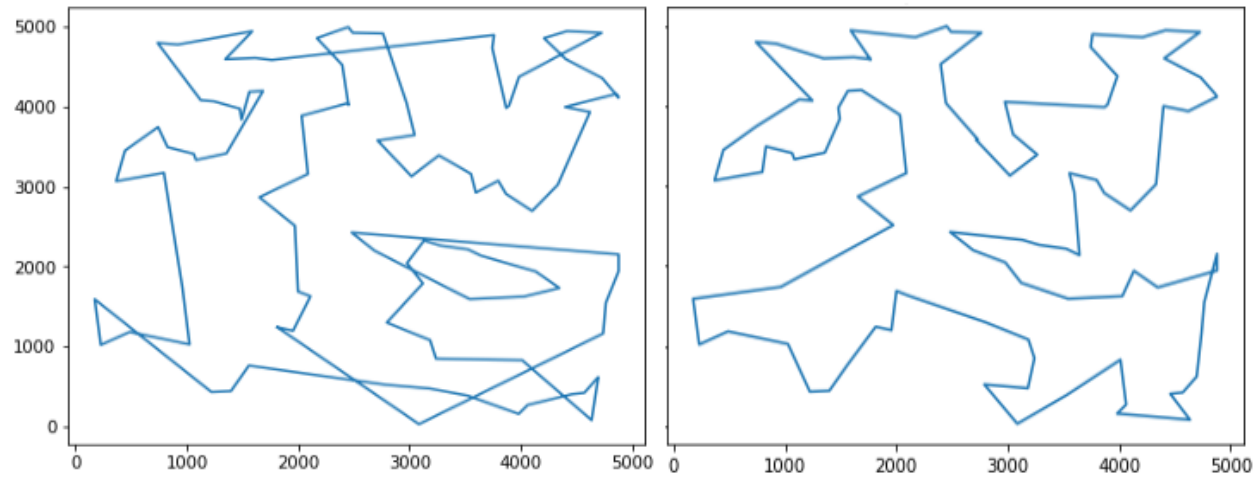

Рисунок 1 — Путь, полученный эвристическим алгоритмом ближайшего соседа (слева) и метаэвристическим алгоритмом имитации отжига (справа) для количества городов $N = 100$. Длины путей соответственно равны 47745 и 38696.

Необходимость использования тех или иных подходов к решению задачи коммивояжёра обусловлена размером исходных данных, доступной производительностью, отведённым временем на выполнение и требуемой точностью результата. Например, в случае использования навигаторов необходима точность на небольшом размере исходных данных при низкой производительности и ограниченном времени. Здесь можно воспользоваться точными алгоритмами.

В случае подбора оптимальных маршрутов доставок товаров наилучшим вариантом является применение эвристических алгоритмов, отличающихся предсказуемостью скорости работы, отсутствием необходимости настройки гиперпараметров и неплохими результатами при невысоком размере исходных данных.

Однако уже при прокладке коммуникационных сетей необходима точность на большом количестве точек при наличии высокой производительности и достаточного времени. А это значит, что стоит обратить внимание на метаэвристические алгоритмы.

Разработанные программы, тесты и полученные результаты доступны по ссылкам:
1) https://github.com/astraszab/TSP-algorithms;
2) https://github.com/dmitriyklebanov/tsp.

Список использованных источников:
1. Алгоритмы. Построение и анализ / Т. Кормен [и др.] ; пер. с англ. — 3-е изд. — М. : Вильямс, 2015. — 1328 с.
2. Скиена, С. Алгоритмы. Руководство по разработке / С. Скиена ; пер. с англ. — 2-е изд. — СПб. : БХВ, 2017. — 760 с.

МОБИЛЬНОЕ ПРИЛОЖЕНИЯ ДЛЯ ИНДИВИДУАЛЬНЫХ ЗАНЯТИЙ СПОРТОМ

Бастун А.Н.

Белорусский государственный университет информатики и радиоэлектроники
г. Минск, Республика Беларусь

Таборовец В.В. – к.т.н.,доцент

Рассматриваются вопросы создания мобильного приложения для фиксации и автоматического анализа результатов контроля данных о физическом состоянии здоровья человека с использованием платформы iOS.

В современном мире вопрос укрепления здоровья человека становиться все более изученным и количество факторов влияющих на него становится все больше. У людей становится все больше носимых устройств, способных собирать данные о состоянии человека в течении дня, таких как телефоны, умные браслеты, часы и т.д. С помощью этих данных можно строить планы тренировок, рекомендации по питанию и т.д. Поскольку стоимость работы персонального тренера, диетолога или другого специалиста является существенной для большинства, либо у людей просто нет на это времени. Использование простого приложения для тренировок и отслеживания текущих показателей может быть большим плюсом в укреплении здоровья, развития общего физического состояния и т.д

Цель разработки является создание мобильного приложения для платформы iOS, способного считывать данные о физическом состоянии пользователя и его активности и предлагать рекомендации в зависимости от целей пользователя. Данные будут считываться в зависимости от имеющихся устройств (телефон, часы и т.д.) и в зависимости от данных разрешений на использование пользователем. Различные международные исследование показали, что постоянная умеренная физическая нагрузка благотворно влияет на работу мозга и сердечно-сосудистой системы, сокращает риски возникновения различных заболеваний [2].

Основной целью приложения является возможность предоставить пользователю более легкий способ следить за своим физическим состоянием и получать максимальную пользу из физических нагрузок тратя минимальное количество времени. Для считывания данных пользователя будет использоваться нативный фреймворк HealthKit, который предоставляет из себя библиотеку различных показателей физического состояния и активностей [1]. Данный фреймворк находится в экосистеме iOS и дает возможность интеграции с различными носимыми устройствами, также содержит различную медицинскую информацию пользователя. Активно используется различными медицинскими учреждениями в Америке для постоянного наблюдения за пациентами[1, 3].

Данные, которые будет использовать приложение:
1) частота пульса;
2) количество шагов сделанных пользователем;
3) количество энергии использованной в состоянии покоя;
4) количество энергии потраченной на активную деятельность;
5) количество времени проведенное в движении (легкой прогулки или более интенсивно);
6) количество потребленной энергии из еды;
7) Опросы пользователей для получение различных данных, которые нет возможности получить с носимых устройств.

Анализ собранных данных производится на основе анализа интенсивности нагрузки (частота пульса и скорость восстановления пульса до нормальных значений), объема нагрузки (количество сделанных шагов, количество времени и энергии потраченное на активную деятельность) и различные опросы и тесты, сделанные пользователем на основе опросников.

Приложение реализует возможность просмотра предлагаемых рекомендаций, выбор и установку целей, личных параметров и других дополнительных параметров, просмотр статистики.

Список использованных источников: