

Санкт-Петербургский государственный университет

**Материалы весенней научно-практической
конференции по вопросам информатики,
математики, механики и астрономии
МАТ-МЕХ. НАУКА 2024**

29 апреля – 3 мая 2024 г.
Санкт-Петербург

Санкт-Петербург
2024

УДК 004; 51; 531/534; 52

ББК 16; 22.1; 22.2; 22.6

М34

*Печатается по рекомендации
кафедры системного программирования
Санкт-Петербургского государственного университета*

**Материалы весенней научно-практической конференции по во-
просам информатики, математики, механики и астрономии
«Мат-мех. Наука 2024».** 29 апреля – 3 мая 2024 г. Санкт-Петербург — СПб.: Издательство ВВМ, 2024. — 230 с.

ISBN 978-5-9651-1568-6

Тематика сборника затрагивает широкий круг актуальных проблем теоретической и прикладной математики, информатики, механики и астрономии. Цели конференции: представление результатов и организация научного общения преподавателей, ведущих учёных и молодых исследователей, апробация результатов студентов и аспирантов, установление и укрепление научных связей между исследовательскими группами.

Для студентов и аспирантов естественно-научных специальностей.

**Вероятностные графические модели,
нечеткие системы, мягкие вычисления и
социокомпьютинг, машинное обучение**



Абрамов Максим Викторович
к.т.н., доцент кафедры информатики

Разработка пакета нейросетевой аппроксимации дифференциальных уравнений DEGANN

Алимов П.Г., СПбГУ, Санкт-Петербург st076209@student.spbu.ru,
Гориховский В.И., СПбГУ, Санкт-Петербург v.gorikhovskii@spbu.ru

Аннотация

Во многих современных областях наук возникает необходимость моделирования динамических систем, и для ускорения вычисления дифференциальных уравнений исследователи прибегают к аппроксимации решения дифференциальных уравнений с помощью нейронных сетей. При этом на текущий момент отсутствует инструментарий, позволяющий быстро пользоваться ся нейронными сетями для аппроксимации.

Данная работа посвящена разработке Python пакета DEGANN для нейросетевой аппроксимации дифференциальных уравнений по заданным требованиям к полученному решению.

Введение

Во многих современных областях наук возникает необходимость моделирования динамических систем.

- В работе [1] авторам требовалось многократное нахождение решения системы дифференциальных уравнений с помощью прямого моделирования методом Монте-Карло (DSMC). Для уменьшения вычислительной сложности расчётов авторы разработали и обучили нейронную сеть, позволяющую аппроксимировать численные решения.
- В исследовании [2] авторы проводили моделирование биологических сетей, содержащих более 10 узлов. Для решения этой задачи была реализована многослойная нейронная сеть, основанная на перцептронах.
- В работе [3] сделана попытка смягчить ограничения традиционного численного подхода к моделированию проводимости ионного канала с помощью применения нейронных сетей.

Несмотря на то, что в решении множества различных задач успешно применяются нейронные сети, оказывается, что не существует универсального инструмента, позволяющего быстро пользоваться таким подходом. Текущие наработки предлагают только решения частных случаев.

- В исследовании 2017-2019г. Raissi et al. были разработаны нейронные сети для аппроксимации решений небольшого набора уравнений в частных производных: Бюргера, Шрёдингера, Навье-Стокса, Аллен-Кана и Кортевега — де Фриза [4].
- Flamant et al. разрабатывают универсальную нейронную сеть, которая может аппроксимировать решение любого дифференциального уравнения [5]. Полученный инструмент не подходит для быстрых серийных вычислений в связи со сложностью самой НС.

Таким образом создание решения, позволяющего автоматически подбирать топологию и параметры обучения нейронной сети для аппроксимации решения заданного дифференциального уравнения, является актуальной.

Архитектура решения

Для создания решения, предназначенного для автоматического поиска подходящей топологии нейронной сети, необходимо создать модуль, позволяющий генерировать настраиваемые НС. Для этого нужно сделать обёртку над существующим инструментом для работы с нейронными сетями или собственную реализацию.

Также необходимо определить методы оптимизации в пространстве топологий нейронных сетей. И для достижения автоматизации должна быть реализована экспертная система, которая автоматически предлагает метапараметры для методов оптимизации, которую в свою очередь возвращают подходящую топологию нейронной сети.

Таким образом решение на высоком уровне должно представлять собой экспертную систему. А экспертная система, в свою очередь, должна представлять собой оболочку над библиотекой генерации нейронных сетей, которая позволяет по заданным параметрам создавать и обучать нейронные сети, осуществлять их сохранение и загрузку в виде набора весов, производить построение исполняемого кода на C++ для проведения серийных вычислений. Общая архитектура решения представлена на рисунке 1 и относится к классу слоистых архитектур.

Эксперименты

Для анализа точности, эффективности и производительности алгоритмов поиска оптимальной топологии, а также определения подходящих параметров алгоритмов поиска были поставлены эксперименты.

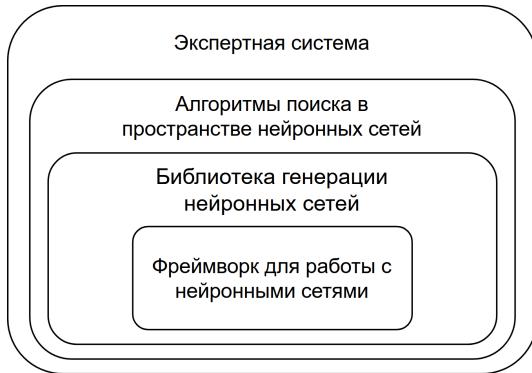


Рис. 1: Общее представление архитектуры пакета DEGANN.

Дизайн экспериментов

Для проведения экспериментов были выбраны семь уравнений:

$$y' + 3y = 0, \quad y(0) = 1 \quad (\text{LF-ODE1})$$

$$y = \ln(1 + x) \quad (\text{Log})$$

$$y'' + 100y = 0, \quad y(0) = 0, \quad y'(0) = 10 \quad (\text{LH-ODE1})$$

$$y = e^{-\frac{(x-0.5)^2}{0.08}} \quad (\text{Gauss})$$

$$y' + \frac{y - 2x}{x + 0.1} = 0, \quad y(0) = 20 \quad (\text{NL-ODE1})$$

$$y' = 0, \quad y(0) = 1 \quad (\text{LF-ODE2})$$

$$y = \frac{\sin(5x) * \log_2(u + 1)}{\sqrt{1 + t}} \quad (\text{Multidim})$$

Результаты экспериментов

Таблица 1 описывает вероятность в процентах получения, в результате обучения, нейронной сети, которая преодолевает порог в 25% по функции потерь MeanAPE, для случайного поиска и метода имитации отжига *SAM_lin_lin*. Можно заметить, что на всех уравнениях, кроме LH-ODE1, Multidim, оба алгоритма дают большую вероятность успешного обучения НС,

при чём случайный поиск даёт результат быстрее, чем метод имитации отжига. Но на LH-ODE1 и Multidim метод имитации отжига даёт вероятность сходимости на порядок больше.

Таблица 1: Таблица с вероятностью успешного обучения нейронной сети на ДУ с порогом 25% по функции потерь MeanAPE

Название уравнения	Случайный поиск			<i>SAM_lin_lin</i>		
	50	150	400	50	150	400
Размер данных						
Log	42.5%	80%	21.9%	57.1%	60.6%	22.2%
Gauss	47.6%	76.9%	83.3%	22.2%	44.4%	35.7%
NL-ODE1	100%	100%	100%	100%	100%	100%
LF-ODE1	90.9%	100%	100%	100%	100%	100%
LF-ODE2	100%	100%	100%	100%	100%	100%
LH-ODE1	0.9%	0%	7.4%	2.1%	0%	8.3%
Multidim	2.6%	<0.1%	26.6%	4.5%	0.2%	26.6%

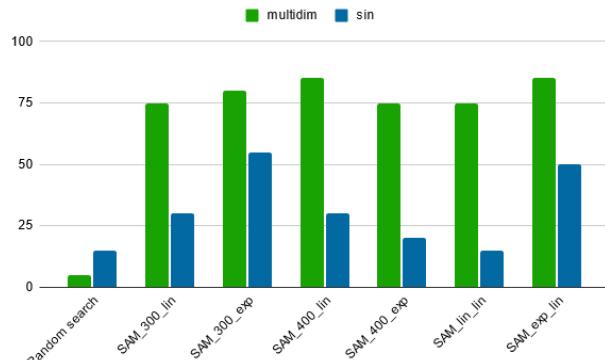


Рис. 2: Вероятность сходимости алгоритмов поиска для уравнений LH-ODE1, Multidim и ограничения 10% по метрике MeanAPE.

На рисунке 2 изображена гистограмма, отражающая вероятность получения результата, подходящего под заданные условия, для алгоритмов случайного поиска и метода имитации отжига. Видно, что метод имитации отжига гарантирует большую вероятность сходимости, чем случайный поиск, и при этом разные конфигурации лучше подходят под разные задачи.

Таким образом, агрегируя выводы, получаем следующие результаты.

- Более чем в 90% случаев случайного поиска достаточно для получения

подходящей топологии нейронной сети.

- При определённых конфигурациях метод имитации отжига сходится на уравнениях (LH-ODE1, Multidim) в 3-4 раза чаще, чем случайный поиск.
- На уравнениях помимо (LH-ODE1, Multidim) для нахождения подходящей топологии нейронной сети алгоритмам требуется обучить менее 100 моделей.
- При увеличении ограничений на требуемое значение функции потерь вероятность сходимости алгоритма случайного поиска убывает быстрее, чем вероятность сходимости метода имитации отжига для поиска оптимальной топологии НС.

Экспертная система

По результатам проведённых экспериментов были выделены возможные метки, описывающие поданную на вход задачу, и в зависимости от набора методов экспертная система проводит конфигурацию конвейера алгоритмов поиска топологии нейронных сетей. Также реализован графический интерфейс над экспертной системой, делающий её более простой в использовании.

Архитектура

По результатам экспериментов были выделены метки, описывающие ограничения на получаемое решение, и общая архитектура представленная на рисунке 3. По меткам строятся ограничения и выбираются алгоритмы и их параметры, которые с наибольшей вероятностью удовлетворят запрос пользователя. Метки представлены в следующем списке.

1. Требуемая точность решения (*precision*).
2. Тип функции, который принимает решение дифференциального уравнения (*type*).
3. Время работы метода feedforward (*predict*) обученной нейронной сети (*work time*).
4. Размер тренировочной выборки данных (*data size*).

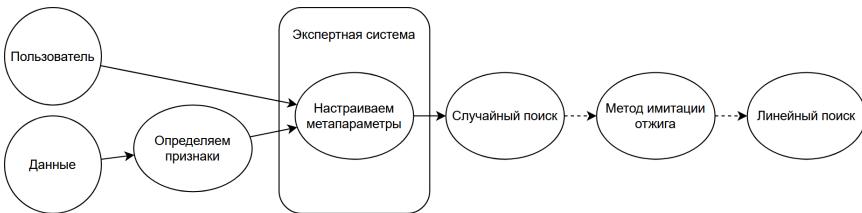


Рис. 3: Поток действий при работе с экспертной системой.

Пример работы

Для валидации работоспособности экспертной системы перед ней были поставлены две задача аппроксимации модели нагруженного гармонического осциллятора с учётом свободных вращений (FHO-FR). Данная модель взята из ВКР Андрея Исакова. Были выставлены следующие требования к решению:

- максимальная точность аппроксимации — для метки *precision* было установлено значение *maximal*;
- скорость аппроксимации не является критическим фактором — для метки *work time* было установлено значение *long*.

Также в качестве подсказок системе были переданы *unknow* и *very small* в качестве значений меток *precision* и *data size* соответственно.

В результате запуска параметризованных экспертной системой алгоритмов поиска была получена обученная нейронная сеть в виде словаря параметров, которую в дальнейшем можно экспортить как в виде параметров, так и в качестве функции на C++. Полученная нейронная сеть содержала три скрытых слоя по 15, 17 и 10 нейронов в каждом соответственно и показала ошибку в 1% по функции потерь MeanAPE на валидационных данных, что говорит о правильном определении топологии НС для данной задачи и положительной оценке работы экспертной системе в том числе и в реальных задачах.

Заключение

В рамках данной работы были получены следующие результаты.

1. Реализованы алгоритмы случайного, линейного поисков и метод имитации отжига для поиска оптимальной топологии в пространстве топологий нейронных сетей.

2. Выполнено экспериментальное исследование, показавшее, что метод имитации отжига на функциях с несколькими независимыми переменными и функциях вида синусоиды получает требуемый результат в 3-4 раза чаще, чем случайный поиск, но при работе обучает в среднем большее количество нейронных сетей и дольше работает.
3. Реализован и опубликован пакет DEGANN¹, включающий в себя библиотеку генерации нейронных сетей, алгоритмы поиска в пространстве топологий нейронных сетей и экспертную систему для автоматического подбора подходящей топологии нейронной сети для аппроксимации заданного дифференциального уравнения.

Список литературы

- [1] Aksanova, Olga A. and Khalidov, Iskander A. Simulation of unstable rarefied gas flows in a channel for different Knudsen numbers // AIP Conference Proceedings 2132, 180009 (2019). <https://aip.scitation.org/doi/abs/10.1063/1.5119667>
- [2] Mao Guo, Zeng Ruigeng, Peng Jintao, Zuo Ke, Pang Zhengbin, Liu Jie. Reconstructing gene regulatory networks of biological function using differential equations of multilayer perceptrons // BMC Bioinformatics 23, 503 (2022). <https://doi.org/10.1186/s12859-022-05055-5>
- [3] Lei Chon Lok, Mirams Gary R. Neural Network Differential Equations For Ion Channel Modelling // Front. Physiol. 12:708944. <https://www.frontiersin.org/articles/10.3389/fphys.2021.708944>
- [4] Raissi M., Perdikaris P., Karniadakis G.E. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations // Journal of Computational Physics, Volume 378, Pages 686-707. <https://www.sciencedirect.com/science/article/pii/S0021999118307125>
- [5] Flamant Cedric, Protopapas Pavlos, Sondak David. Solving Differential Equations Using Neural Network Solution Bundles // <https://arxiv.org/abs/2006.14372>

¹Доступен по ссылке: <https://pypi.org/project/degann/>.

Целочисленное квантование для вывода при глубоком обучении

Владимирова Э.В., студент кафедры информатики СПбГУ, техник-программист
 ООО «Системы компьютерного зрения», Санкт-Петербург
 st069281@student.spbu.ru

Аннотация

В статье рассмотрена новая реализация метода Quant-Noise для нейросетей ResNet-20 и EfficientNet-B3 ввиду отсутствия оригинальной и проведено сравнение эффективности обучения нейросети для подавления шума CBDNet с использованием методов LSQ, LSQ+ и QSin.

Введение

Практическое использование нейросетей в особенности на мобильных устройствах приводит к необходимости сжатия моделей по памяти и вычислительной сложности. Одним из зарекомендовавших себя методов является квантование весов сети и вычислений.

В рамках проекта «Discrete Optimization for Accurate Low-bit Quantization» лаборатории им. П. Л. Чебышёва по разработке нового метода квантования нейросетей для обработки изображений в форматы int8, int4 и int2 одной из промежуточных задач была подготовка данных для сравнения с результатами существующих методов на общем наборе архитектур нейросетей и данных.

В статье представлены новые результаты обучения нейросети CBDNet [3] на наборе изображений SIDD с использованием существующих методов квантования LSQ [7], LSQ+ [8] и QSin [10] и их сравнение. Также представлены результаты собственной реализации метода Quant-Noise [9] на нейросетях ResNet-20 [1] и EfficientNet-B3 [6] с наборами изображений CIFAR-10 и ImageNet в конфигурации w4a4 (квантование весов и активаций нейросети до типа int4).

Обучение с учётом квантования

Операция квантования вещественного числа представима функцией [5]:

$$x_q = \text{quantize}(x, b, s, z) = \text{clip}(\text{round}(s \cdot x + z), -2^{b-1}, 2^{b-1} - 1),$$

где x — исходное вещественное значение, b — количество бит в целочисленном типе, $z = -\text{round}(\beta \cdot s) - 2^{b-1}$ — нулевая точка квантования, масштаб (шаг) квантования —

$$s = \frac{2^b - 1}{\alpha - \beta},$$

функция обрезания —

$$\text{clip}(x, l, u) = \begin{cases} l, & x < l \\ x, & x \in [l; u] \\ u, & x > u \end{cases}$$

Для весов рассматривается симметричное квантование, $z = 0$. Обучение квантованной сети методом градиентного спуска без дополнительных модификаций вызывает затруднения из-за кусочной постоянности функции квантования. Эта проблема решается либо гладкой регуляризацией функции квантования (например, QSin) [10], либо прямолинейным оцениванием (Straight-through Estimator, STE) [2, 7, 8], игнорирующим квантование на обратном проходе по сети и доопределяющим производную:

$$\tilde{x} = \text{dequantize}(\text{quantize}(x, b, s), b, s).$$

Эти подходы обучения с учётом квантования (Quantization Aware Training, QAT), позволяют обучать квантованную сеть обычными методами.

Метрики

Для задач классификации (ResNet-20 и EfficientNet-B3) используется метрика точность (accuracy). Для задачи подавления шума используются метрики сходства изображений: пиковое отношение сигнала к шуму (PSNR) и индекс структурного сходства (SSIM).

Метод рандомизированного добавления шума квантования

Описание задачи

Метод рандомизированного добавления шума квантования (Quantization Noise, Quant-Noise) [9] является модификацией QAT и заключается в применении симуляции эффекта квантования к рандомизированной выборке определённого процента весов из всего набора. Доля квантуемых весов — гипер-

параметр метода. Выборка генерируется индивидуально для каждого прямого прохода в ходе обучения нейросети. Активации в дополненном процессе обучения не затрагиваются. Авторами приводятся численные доказательства большей эффективности описанного подхода в сравнении с оригинальным QAT для нейросетей из двух различных областей глубокого машинного обучения: обработки естественного языка и классификации изображений, из которых в рамках проекта актуальна вторая. Вместе с тем имеют место следующие проблемы:

- официальная реализация алгоритма не завершена для нейросетей, решающих задачи компьютерного зрения. Репозиторий [4], указанный в статье как источник кода, содержит не адаптированную к анализу цветных изображений и не обеспечивающую сходимость обучения квантованной нейросети версию алгоритма;
- подходящие техническому заданию проекта результаты представлены только для нейросети EfficientNet-B3 и датасета ImageNet, что усложняет сравнение эффективности Quant-Noise с остальными методами.

Таким образом, работа с алгоритмом Quant-Noise потребовала независимой реализации.

Реализация

В официальном репозитории [4] симуляция квантования организована заменой свёрточных и линейных слоёв нейросети на аналогичные исходным слои с дополнительными параметрами. Применением описанных классовых преобразований к ResNet-20 и восстановлением работоспособности алгоритма была экспериментально выявлена непригодность оригинальной реализации: продемонстрированные при обучении с набором конфигураций результаты значительно уступают заявленным в статье и полученным для реализации алгоритма на основе метода QAT.

В качестве базы для собственной реализации алгоритма Quant-Noise использована реализация метода QAT, созданная в рамках проекта с использованием PyTorch. Выборка весов, имеющих вид многомерного тензора, производится генерацией случайной бинарной маски по распределению Бернулли. Ввиду сложности архитектуры и длительности обучения целевых нейросетей для разработки были взяты следующие комбинации наборов данных и нейросетей на каждом этапе:

1. **запуск алгоритма:** датасет — MNIST, нейросеть — встроенный шестисторонний классификатор;

2. **отладка и тестирование алгоритма:** датасет — CIFAR-10, нейросеть — ResNet-20;
3. **сравнение результатов со статьёй:** датасет — ImageNet, нейросеть — EfficientNet-B3.

Созданная в ходе проекта стабильная реализация Quant-Noise подтвердила ожидаемое улучшение точности по сравнению с QAT на нейросети ResNet-20 и датасете CIFAR-10 (Таблица 1). Тем не менее, обучение квантованной с Quant-Noise нейросети длилось в 1.5 раза дольше обучения квантованной с QAT и в 9 раз дольше обучения исходной вещественнозначной. Внесением оптимизаций в код и повышением размера батча разрыв между временем работы реализаций двух методов был сокращён.

Конфигурация	QAT	Quant-Noise (prob=0.5)	Quant-Noise (prob=0.125)
w8a8	91.76	91.88	91.87
w4a4	89.27	90.54	89.8

Таблица 1: Точность (accuracy) ResNet-20 на CIFAR-10 при обучении с методами QAT и Quant-Noise w4a4

На целевых нейросети EfficientNet-B3 и датасете ImageNet было выявлено существенное замедление обучения внедрённым Quant-Noise, в итоге частично нейтрализованное изменением способа генерации и кэшированием бинарных масок. При проверке гипотез о причинах и возможных решениях проблемы реализация была разбита на две в зависимости от наличия опции хранения масок. Профилирование реализаций QAT и Quant-Noise показало сопоставимость требуемых вычислительных и временных ресурсов для обучения нейросети EfficientNet-B3 с двумя методами (Таблица 2).

	floating point	QAT	Quant-Noise (prob=0.5)
Время на 1 эпоху (ч:мин)	1:55	6:05	6:28
Утилизация GPU (%)	91—93	57—67	57—68
Память GPU (%)	63.1	76.4	93.6

Таблица 2: Время 1 эпохи и загрузка GPU при обучении EfficientNet-B3 на ImageNet методами QAT и Quant-Noise w4a4

Также был проведён подбор наиболее оптимальных по времени и точности работы квантованной нейросети гиперпараметров обучения, с которыми

были достигнуты показатели точности в Таблице 3.

Модель	Accuracy
Данные авторов floating-point модели [6]	82.008
Валидация floating-point модели	82.032
QAT	65.324
Quant-Noise	77.154

Таблица 3: Точность (accuracy) EfficientNet-B3 на ImageNet при обучении методами QAT и Quant-Noise w4a4

Оценка показателей обучения целевых методов

Оценка эффективности разрабатываемого в рамках проекта алгоритма целочисленного квантования строится на сравнении следующих показателей:

1. время обучения (время прохождения 1 эпохи);
2. скорость обучения (значения целевых метрик после 80 эпох обучения);
3. требования обучения к памяти графического процессора (максимальный доступный размер батча);

для нейросетей, квантованных с помощью лучшего по большинству указанных характеристик из выбранных ориентировочных методов: LSQ [7], LSQ+ [8] и QSin [10] — для конфигурации w4a4. В рамках поставленной задачи в качестве тестовой комбинации взяты нейросеть CBDNet [3] и 2 поднабора набора данных SIDD: Small и Medium. Эксперименты проведены на графическом профессоре NVIDIA Tesla A10.

Показатели в Таблицах 4, 5 получены при размере батча — 30 изображений. Максимальный размер батча для рассматриваемых методов — 120 изображений, из чего допустимо предположить примерное равенство требуемых для обучения ресурсов памяти GPU.

Датасет	LSQ	LSQ+	QSin
SIDD Small	2:57	3:31	4:08
SIDD Medium	5:54	7:02	8:16

Таблица 4: Время 1 эпохи обучения CBDNet методами LSQ, LSQ+, QSin w4a4 (мин:с)

Датасет	LSQ		LSQ+		QSin	
	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
SIDD Small	35.135	0.815	35.221	0.816	25.024	0.371
SIDD Medium	35.531	0.828	35.488	0.828	25.099	0.372

Таблица 5: Метрики CBDNet при обучении методами LSQ, LSQ+, QSin w4a4

Поскольку результаты LSQ+ и LSQ близки и LSQ лучше на большом наборе данных, то для визуального сравнения был выбран LSQ. На рисунке 1 видно, что, несмотря на относительно высокие значения метрик, квантование приводит к заметным визуальным артефактам.

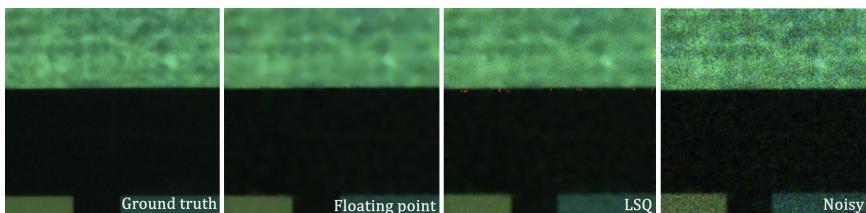


Рис. 1: Визуальное сравнение CBDNet при обучении методом LSQ w4a4

Заключение

В работе приведены итоги реализации метода квантования Quant-Noise для нейросети ResNet-20 на датасете CIFAR-10 и EfficientNet-B3 на датасете ImageNet, предложены способы повышения эффективности по использованию ресурсов памяти GPU и времени реализации. Реализация Quant-Noise в рамках проекта для квантования нейросети в формат int4 улучшает accuracy на 1.27% для ResNet-20 на CIFAR-10 и 11.83% EfficientNet-B3 на ImageNet по сравнению с QAT, чем подтверждает результаты оригинальной статьи.

Также проведено сравнение времени, скорости и требований к памяти для обучения нейросети CBDNet на поднаборах Small и Medium набора данных SIDD для методов квантования LSQ, LSQ+ и QSin. В качестве главного конкурирующего метода для дальнейшего сравнения берётся LSQ.

Список литературы

- [1] Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun. Deep Residual Learning for Image Recognition. — arXiv, 2015
- [2] Benoit Jacob, Skirmantas Kligys, Bo Chen. Quantization and Training of Neural Networks for Efficient Integer-Arithmetic-Only Inference. — arXiv, 2017
- [3] Shi Guo, Zifei Yan, Kai Zhang et al. Toward Convolutional Blind Denoising of Real Photographs. — arXiv, 2019
- [4] Fairseq. Training with Quantization Noise for Extreme Model Compression, 2020. — URL: https://github.com/facebookresearch/fairseq/tree/main/examples/quant_noise. (accessed: 2023-05-29)
- [5] Hao Wu, Patrick Judd, Xiaojie Zhang, et al. Integer Quantization for Deep Learning Inference: Principles and Empirical Evaluation. — arXiv, 2020
- [6] Mingxing Tan, Quoc V. Le. EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. — arXiv, 2020
- [7] Steven K. Esser, Jeffrey L. McKinstry, Deepika Bablani et al. Learned Step Size Quantization. — arXiv, 2020
- [8] Yash Bhalgat, Jinwon Lee, Markus Nagel et al. LSQ+: Improving low-bit quantization through learnable offsets and better initialization. — arXiv, 2020
- [9] Angela Fan, Pierre Stock, Benjamin Graham et al. Training with Quantization Noise for Extreme Model Compression. — arXiv, 2021
- [10] Kirill Solodskikh, Vladimir Chikin, Ruslan Aydarkhanov et al. Towards Accurate Network Quantization with Equivalent Smooth Regularizer. — European Conference on Computer Vision (ECCV), 2022

Внедрение оптимизационных алгоритмов в интерфейс фреймворка Streamlit¹

Егоров П.А., СПбГУ, Санкт-Петербург st068715@student.spbu.ru

Аннотация

В ходе работы были реализованы на языке Python и интегрированы в приложение machine-learning-ui оптимизационные алгоритмы AdaMod, MADGRAD, RAdam, Apollo, AdaHessian, LARS, LAMB. В качестве апробации алгоритмов была решена задача расчета поуровневых коэффициентов скорости колебательных энергообменов с применением нейросетевого подхода.

Введение

Streamlit — это открытый Python фреймворк, который позволяет создавать интерактивные веб-приложения для анализа данных и машинного обучения. Фреймворк имеет понятный интуитивный интерфейс и совместим со множеством библиотек, такими как Pandas, Matplotlib, TensorFlow и другими.

Необходимой частью машинного обучения являются оптимизационные алгоритмы, поскольку они позволяет находить оптимальные параметры моделей, минимизировать функцию потерь и повышать точность предсказаний. В нейронных сетях, как правило, используют алгоритмы, основанные на вычислении градиента или гессиана оптимизируемой функции.

На языке Python были реализованы и интегрированы в приложение machine-learning-ui, разработанное на кафедре гидроаэромеханики в рамках исследовательского проекта «Машинное обучение в задачах неравновесной аэромеханики», следующие оптимизаторы: AdaMod [1], MADGRAD [2], RAdam [3], Apollo [4], AdaHessian [5], LARS [6], LAMB [7]. Основным инструментом для создания алгоритмов является библиотека TensorFlow. Для внедрения оптимизаторов в приложение были реализованы виджеты с учетом логики Streamlit, поскольку приложение создано с использованием данного фреймворка.

Детали реализации

В качестве основного инструмента для реализации оптимизационных алгоритмов была использована библиотека TensorFlow. Таким образом, все оптимизаторы должны наследоваться от базового класса

¹M1_2021 - 3: Машинное обучение в задачах неравновесной аэромеханики: 2023 г. этап 3

`tensorflow.keras.optimizers.Optimizer`. При создании алгоритмов необходимо переопределить несколько методов базового класса. В конструкторе `__init__()` определяются гиперпараметры модели, такие как скорость обучения, коэффициенты затухания и т.д. Инициализация переменных оптимизатора, например, импульса или экспоненциального скользящего среднего градиентов осуществляется в методе `build()`, принимающем на вход список переменных модели `var_list`. Основная логика алгоритма реализуется в методе `update_step()`, который в качестве аргументов принимает переменную и градиент. Результат работы данного метода обновляет веса нейронной сети. Для сохранения конфигурации оптимизатора и последующего запуска реализуется метод `get_config()`, возвращающий словарь со значениями гиперпараметров оптимизационного алгоритма.

В оптимизаторе AdaHessian, помимо переменных и градиентов, которые автоматически вычисляются в TensorFlow, необходимо хранить информацию об аппроксимации матрицы Гессе, которая рассчитывается при дифференцировании градиентов функции потерь. Поэтому при реализации данного алгоритма были переопределены и другие методы класса `Optimizer`, включая такие, как `compute_gradients()`, `apply_gradients()`, `minimize()` и т.д.

Основное приложение написано с использованием фреймворка Streamlit. В связи с этим для интеграции оптимизационных алгоритмов были реализованы виджеты с учетом логики Streamlit. Классы виджетов содержат в себе описания, диапазоны значений и строковые форматы гиперпараметров оптимизаторов.

Диаграммы реализованных оптимизаторов выглядят следующим образом:

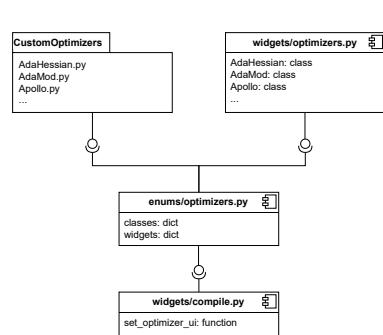
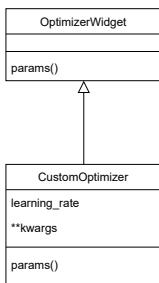
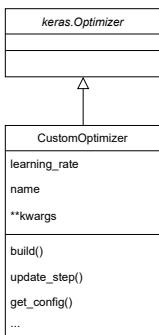


Рис. 1: Диаграммы классов оптимизаторов и виджетов

Рис. 2: Диаграмма компонентов

Тестирование алгоритмов

Условия эксперимента

Для тестирования работы оптимизационных алгоритмов с помощью библиотек NumPy и Pandas был сгенерирован набор данных, равномерно распределенных на отрезке $[0, 1)$, размерности 10000×50 с добавлением шума, имеющего стандартное нормальное распределение. В качестве тестовой модели была использована нейронная сеть с тремя входными слоями, каждый из которых соединен с отдельным полно связанным слоем. Далее расположен слой конкатенации, связывающий три предыдущих слоя, а за ним еще один полно связанный слой. Он, в свою очередь, разделяется еще на два полно связанных слоя, которые являются выходными слоями модели.

Помимо обучения с использованием реализованных оптимизационных методов, в ходе эксперимента модель была обучена с оптимизатором Adam, доступным в библиотеке TensorFlow. В таблице 1 представлены гиперпараметры алгоритмов, использованные в процессе эксперимента.

Оптимизатор	Гиперпараметры
AdaMod	$lr = 0,001, \beta_1 = 0,9, \beta_2 = 0,999, \beta_3 = 0,995$
RAdam	$lr = 0,001, \beta_1 = 0,9, \beta_2 = 0,999$
MADGRAD	$lr = 0,01, \text{momentum} = 0,9$
Apollo	$lr = 0,01, \beta = 0,9, \text{weight_decay} = 0,001$
AdaHessian	$lr = 0,01, \beta_1 = 0,9, \beta_2 = 0,999, \text{weight_decay} = 0,001$
LARS	$lr = 0,01, \beta = 0,9$
LAMB	$lr = 0,001, \beta_1 = 0,9, \beta_2 = 0,999$
Adam	$lr = 0,001, \beta_1 = 0,9, \beta_2 = 0,999$

Таблица 1: Оптимизаторы и их гиперпараметры

Метрики

В качестве функции потерь использовалась среднеквадратичная ошибка (Mean Squared Error, MSE). Она имеет свойство штрафовать большие ошибки сильнее, нежели другие метрики за счет того, что результат ошибки возводится в квадрат.

Также для оценки качества обученных моделей была использована метрика — логарифм гиперболического косинуса ошибки предсказания. Данная

метрика менее чувствительна к выбросам, в отличие от среднеквадратичной и средней абсолютной ошибок, а исходные данные были зашумлены.

Результаты обучения

В таблице 2 представлены значения времени обучения модели с различными оптимизаторами.

Алгоритм	Время обучения, с	Число эпох
AdaMod	$15,8 \pm 1,5$	22
MADGRAD	$19,6 \pm 1,3$	24
RAdam	$17,9 \pm 0,4$	23
Apollo	$12,6 \pm 1,4$	17
AdaHessian	$24,1 \pm 0,6$	30
LARS	$21,5 \pm 0,4$	30
LAMB	$15,8 \pm 1,5$	21
Adam	$16,36 \pm 0,8$	21

Таблица 2: Время обучения алгоритмов

Быстрее всех обучилась модель с оптимизатором Apollo — приблизительно 12,5 секунд. Примерно одинаковое время обучения показали алгоритмы первого порядка AdaMod, LAMB и Adam — около 16 секунд. RAdam превзошел по скорости MADGRAD, который оказался одним из самых долгообучаемых среди алгоритмов первого порядка, за исключением оптимизатора LARS со средней скоростью обучения 21,5 секунд. Медленнее всех обучилась модель с оптимизатором AdaHessian — примерно 24 секунды.

Расчет поуровневых коэффициентов скорости колебательных энергообменов

В качестве апробации была решена задача расчета коэффициентов скорости колебательных энергообменов. Для решения данной задачи были представлены наборы данных VT-обменов между молекулой O_2 и частицами партнерами по столкновению (O_2, O, Ar), рассчитанные с помощью модели нагруженного гармонического осциллятора с учетом свободных вращений. Размерности наборов данных составили 525 элементов для всех пар частиц. Элементы наборов данных состоят из номеров уровней перехода, тем-

ператур и вычисленных коэффициентов. В работе [8] регрессионной моделью аппроксимировали коэффициенты отдельно для каждого уровня только по значениям температуры, но в этом случае размерность набора данных составляет всего 14 элементов. Данный подход нецелесообразно использовать при обучении нейронной сети. Поэтому нейросетевые модели были обучены по всему набору данных. Предсказание коэффициентов осуществляется по двум переменным — уровню перехода и температуре.

Для обучения была использована пятислойная модель нейронной сети. В первом и последнем полно связанных слоях использовалась тождественная функция активации для обработки выходного значения. В центральном полно связном слое была применена сигмоидная функция активации, поскольку все данные больше нуля, а также для добавления нелинейности в модель. Для увеличения производительности и борьбы с переобучением между полно связными слоями были добавлены слои пакетной нормализации (Batch Normalization Layer). В качестве функции потерь была выбрана среднеквадратичная ошибка.

Поскольку значения коэффициентов малы, для улучшения качества предсказаний они были предварительно нормализованы — умножены на 10×10^{18} и преобразованы до диапазона $[0, 1]$ с помощью MinMaxScaler:

$$\hat{x}_i = \frac{x_i - x_{min}}{x_{max} - x_{min}}$$

где x_i - значение i -го элемента признака, x_{min} и x_{max} - минимальное и максимальное значения признака.

В таблице 3 представлены значения ошибок и времени обучения для всех наборов данных после масштабирования.

Алгоритм	O_2-O		O_2-Ar		O_2-O_2	
	MSE	Время, с	MSE	Время, с	MSE	Время, с
AdaMod	0,004	6,23	0,117	4,31	0,015	4,44
MADGRAD	0,223	4,76	0,256	4,68	0,168	4,57
RAdam	0,072	6,65	0,069	5,04	0,026	4,94
Apollo	0,261	2,48	0,300	2,85	0,174	2,46
AdaHessian	0,129	5,92	0,087	5,86	0,114	5,85
LARS	0,007	4,43	0,025	4,38	0,029	4,43
LAMB	0,027	4,47	0,063	4,78	0,036	4,67
Adam	0,100	4,23	0,338	3,11	0,352	3,42

Таблица 3: Результаты обучения на преобразованных данных

На рисунке 3 представлены предсказания коэффициентов моделью с оптимизатором AdaMod для задачи O_2-O_2 взаимодействий при одноквантовом переходе с 32 на 31 уровень.

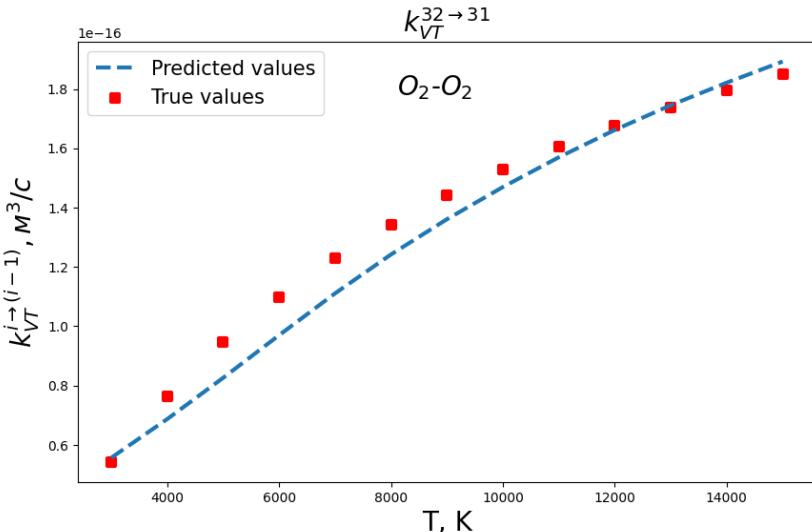


Рис. 3: $k_{VT}^{32 \rightarrow 31}$ O_2-O_2 взаимодействий

Заключение

В ходе работы на языке Python с использованием библиотеки TensorFlow были реализованы и интегрированы в приложение machine-learning-ui оптимизационные алгоритмы: AdaMod, RAdam, MADGRAD, Apollo, AdaHessian, LARS, LAMB. Была проведена апробация алгоритмов. Модель нейросети со всеми оптимизаторами сошлась в течение 30 эпох.

Также была решена задача расчета поуровневых коэффициентов скорости колебательных энергообменов для пар частиц (O_2-O , O_2-Ar , O_2-O_2) с применением нейросетевого подхода. Была подобрана подходящая топология модели для текущей задачи. Обучение модели с различными оптимизационными алгоритмами заняло менее 7 секунд для каждого набора данных с учетом масштабирования. Значение ошибки при валидации реализованных оптимизаторов составило от 0,3 до 0,004, а средняя ошибка после обратного масштабирования оказалась равной — $1,19 \times 10^{-17}$. Предсказанные значения оказались близки к истинным.

Список литературы

- [1] Jianbang Ding, Xuancheng Ren, Ruixuan Luo, Xu Sun. An Adaptive and Momental Bound Method for Stochastic Learning. // arXiv:1910.12249, 2019
- [2] Aaron Defazio, Samy Jelassi. Adaptivity without Compromise: A Momentumized, Adaptive, Dual Averaged Gradient Method for Stochastic Optimization. // arXiv:2101.11075, 2021
- [3] Liyuan Liu, Haoming Jiang, Pengcheng He, Weizhu Chen, Xiaodong Liu, Jianfeng Gao, Jiawei Han. On the Variance of the Adaptive Learning Rate and Beyond. // arXiv:1908.03265, 2021
- [4] Xuezhe Ma. Apollo: An Adaptive Parameter-wise Diagonal Quasi-Newton Method for Nonconvex Stochastic Optimization. // arXiv:2009.13586, 2021
- [5] Zhewei Yao, Amir Gholami, Sheng Shen, Mustafa Mustafa, Kurt Keutzer, Michael W. Mahoney. ADAHESSIAN: An Adaptive Second Order Optimizer for Machine Learning. // arXiv:2006.00719, 2021
- [6] Yang You, Igor Gitman, Boris Ginsburg. Large Batch Training of Convolutional Networks. // arXiv:1708.03888, 2017
- [7] Yang You, Jing Li, Sashank Reddi, Jonathan Hseu, Sanjiv Kumar, Srinadh Bhojanapalli, Xiaodan Song, James Demmel, Kurt Keutzer, Cho-Jui Hsieh. Large Batch Optimization for Deep Learning: Training BERT in 76 minutes. // arXiv:1904.00962, 2020
- [8] Исаков Андрей Алексеевич, Гориховский Вячеслав Игоревич, Мельник Максим Юрьевич. Модели регрессии для расчёта поуроневых коэффициентов скорости колебательных энергообменов. // ВЕСТНИК САНКТ-ПЕТЕРБУРГСКОГО УНИВЕРСИТЕТА. МАТЕМАТИКА. МЕХАНИКА. АСТРОНОМИЯ, 2024

Дистилляция диффузионных моделей для создания 3D контента

Ельцов Д.А., СПбГУ, Санкт-Петербург sthfaceless@gmail.com

Аннотация

В данной работе исследуется дистилляция диффузионных моделей для генерации высококачественных 3D-ассетов, что актуально для видеоигр, образования и электронной коммерции. Современные методы сталкиваются с проблемами качества данных и разнообразия объектов. Сравнивая наборы данных и оценивая прямую генерацию и дистилляцию знаний, исследование подчеркивает преимущества мета-дистилляции. Предложенный двух-модельный подход улучшает сходимость моделей и визуальное качество. Основные результаты включают повышение 3D согласованности и визуального качества с помощью моделей MVDream и Stable Diffusion.

Введение

Создание качественного 3D контента важно для таких областей, как видеоигры, образование, архитектура и дизайн. В видеоиграх высококачественные объекты окружения создаются командами профессионалов в течение многих лет. В образовании технологии дополненной реальности делают изучение сложных концептов более интерактивным и наглядным. Архитекторы и дизайнеры используют 3D инструменты для создания макетов будущих зданий и комнат.

Основной проблемой в создании генеративных моделей для 3D контента является недостаток качественных данных. Набор данных LAION [8] для 2D изображений содержит около 5 миллиардов объектов с описаниями, после фильтрации которых всё равно остаётся необъятное число изображений. В то время как крупнейший 3D набор данных ObjaverseXL [1] включает всего 10 миллионов объектов, многие из которых простые и низкого качества. Поэтому современные подходы к генерации 3D контента часто используют предобученные 2D модели генерации изображения по тексту из-за их высокой вариативности.

Первый подход включает дообучение моделей для создания нескольких видов объекта и их последующей реконструкции, что быстро, но часто приводит к низкому качеству мешей. Этот процесс можно визуализировать на Рисунке 1.

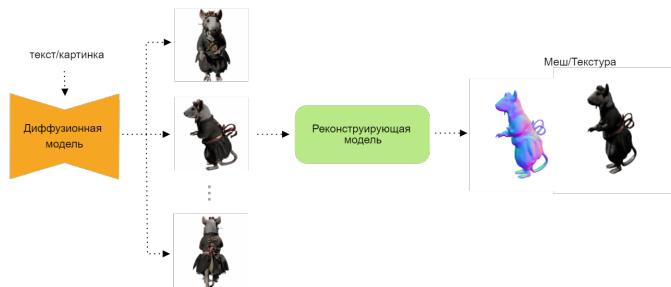


Рис. 1: Прямая генерация 3D объектов с использованием 2D моделей.

Второй подход, являющийся основным объектом исследования, включает инициализацию простым 3D представлением, рендеринг и последующую диффузионную обработку. Этот метод позволяет достичь высокой вариативности и качества рендеринга, зависящих от входного описания или картинки. Однако, он требует значительных вычислительных ресурсов и времени.

Для решения проблемы длительной генерации недавно был предложен оптимизированный подход с использованием дополнительного генератора, который по текстовому описанию создает 3D представление для рендеринга и последующей оптимизации (мета-дистилляция). Этот подход основан на идеи, что многие параметры можно обобщить и переиспользовать, что значительно уменьшает среднее время генерации на один объект и улучшает обобщаемость модели. Это делает данный подход наиболее перспективным для исследования. Данный подход проиллюстрирован на Рисунке 2.

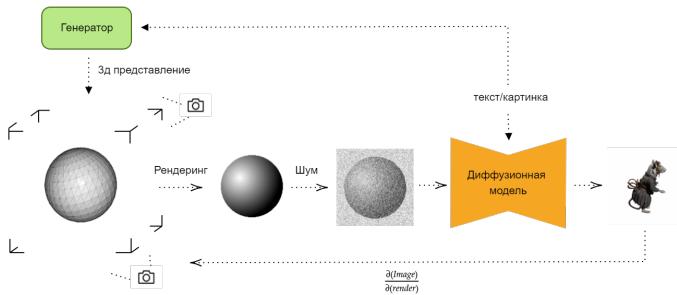


Рис. 2: Мета диффузионная обработка для генерации 3D объектов.

Обзор

На данный момент мета-дистилляция является относительно новой идеей, и хороших статей в популярных журналах пока немного. Основные работы можно упорядочить по увеличению числа текстовых описаний объекта (промптов) и качеству диффузионной модели.

1. Att3D [4]: Базовая работа по мета-дистилляции представлена. В качестве генератора используется обычная полно связная сеть, а 3D представлена в виде неявного объема [5] в низком разрешении.
2. АТоМ [6]: представлен улучшенный генератор на основе трансформера. Первый этап включает неявный объем, а второй - дифференцируемый меш [?]. Однако, диффузионная модель не учитывает 3D согласованность объектов и работает в низком разрешении, что приводит к размытию объектов. В данной работе используется 415 промптов.

Данная работа сосредоточена на качественном улучшении второго подхода с перспективой распространить полученные результаты на большее число промптов. Однако, ни одна из работ пока не имеет открытой программной реализации, поэтому все сравнительные результаты получены с помощью собственной реализации ¹.

Результаты

Реализация базового решения

Согласно оригинальной статье, в качестве диффузионной модели используется диффузионная модель в пиксельном пространстве RGB. В качестве 3D презентации взят объемный рендеринг [5]. Полученные результаты 3 согласованы с результатами статьи. Основная проблема так называемый Janus face (лицо спереди и лицо на спине), так как взятая 2D модель не может учитывать 3D согласованность. Поскольку модель не использует дополнительных автобонкодеров, а работает в пиксельном пространстве, то имеет низкое разрешение из-за вычислительной емкости, что отражается на качестве объектов. Из плюсов можно выделить простоту и хорошее соответствие промпту.

¹GitHub реализация.

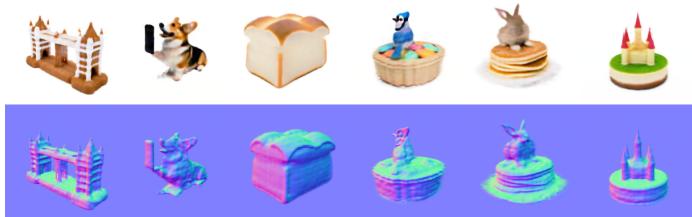


Рис. 3: Результаты базового подхода на основе АТоМ.

Улучшение 3D согласованности

Для улучшения 3D согласованности была взята модель MVDream [9], дообученная с 2D диффузионной модели на 3D данных из Objaverse. Проблема Janus face исчезла и практически полностью решается проблема 3D согласованности объектов, но появились новые проблемы: ухудшение визуального качества, соответствия промпту и появление точечных артефактов, которые являются отрицательным эффектом дообучения на низкокачественных 3D данных.

Добавление второй модели

Успешным экспериментом стало добавление в процесс оптимизации второй модели — оригинальной Stable Diffusion [2] с небольшим весом, что позволило улучшить визуальное качество объектов, сохранило 3D согласованность и значительно уменьшило количество точечных артефактов. Минусом стала увеличение времени на одну итерацию примерно в 1.5 раза, что решается следующим результатом.

Детерминированное расписание шума

Следующим успешным результатом стала адаптация подхода DreamTime [3] с оптимизации одного объекта на режим мета-дистилляции. Основная идея заключается в переходе на правильное детерминированное снижение уровня шума в обучении взамен привычного случайного уровня. Данный подход улучшил общую реалистичность изображений и ускорил сходимость примерно в 2 раза. Однако возникла проблема со сходимостью отдельных объектов, которые не успели получить нужное число обновлений для геометрии на соответствующих уровнях шума. В результате чего было принято решение

начинать обучение со случайного шума и поэтапно переходить в детерминированный вариант для конечных результатов.

Визуальные результаты последовательного применения всех подходов показаны на Рисунке 4.

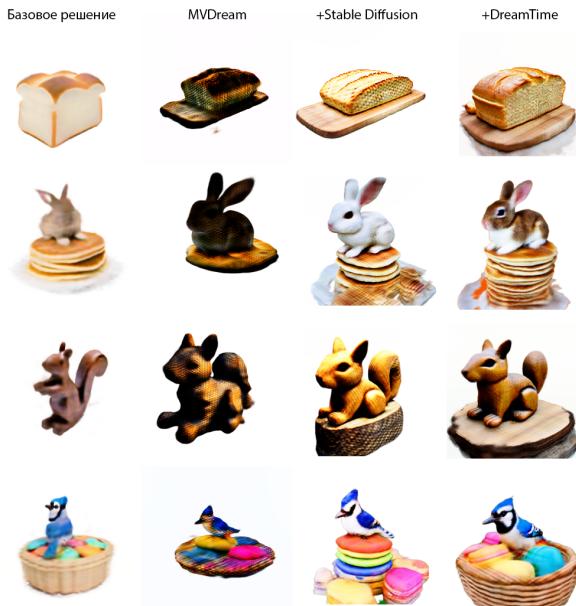


Рис. 4: Визуальные результаты.

Заключение

В результате данной работы был получен ряд успешных экспериментов, значительно увеличивающий 3D согласованность и визуальное качество в сравнении с базовым подходом на основе АТоМ.

Наиболее перспективные направления для дальнейшего развития включают:

1. Оптимизацию архитектуры генератора для более эффективного использования ресурсов и улучшения соответствия промпту.
2. Дальнейшее улучшение визуального качества и детализации объектов путем перехода на дифференцируемый меш рендеринг.

3. Распространение полученных результатов на большие наборы промптов для получения обобщаемой мета модели.

Список литературы

- [1] Deitke, M., Liu, R., Wallingford, M., Ngo, H., Michel, O., Kusupati, A., Fan, A., Laforte, C., Voleti, V., Gadre, S.Y. and VanderBilt, E., 2024. *Objaverse-xl: A universe of 10m+ 3d objects*. Advances in Neural Information Processing Systems, 36.
- [2] Esser, P., Kulal, S., Blattmann, A., Entezari, R., Müller, J., Saini, H., Levi, Y., Lorenz, D., Sauer, A., Boesel, F. and Podell, D., 2024. *Scaling rectified flow transformers for high-resolution image synthesis*. arXiv preprint arXiv:2403.03206.
- [3] Huang, Y., Wang, J., Shi, Y., Qi, X., Zha, Z.J. and Zhang, L., 2023. *DreamTime: An Improved Optimization Strategy for Text-to-3D Content Creation*. arXiv preprint arXiv:2306.12422.
- [4] Lorraine, J., Xie, K., Zeng, X., Lin, C.H., Takikawa, T., Sharp, N., Lin, T.Y., Liu, M.Y., Fidler, S. and Lucas, J., 2023. *Att3d: Amortized text-to-3d object synthesis*. In Proceedings of the IEEE/CVF International Conference on Computer Vision (pp. 17946-17956).
- [5] Mildenhall, B., Srinivasan, P.P., Tancik, M., Barron, J.T., Ramamoorthi, R. and Ng, R., 2021. *Nerf: Representing scenes as neural radiance fields for view synthesis*. Communications of the ACM, 65(1), pp.99-106.
- [6] Qian, G., Cao, J., Siarohin, A., Kant, Y., Wang, C., Vasilkovsky, M., Lee, H.Y., Fang, Y., Skorokhodov, I. and Zhuang, P., 2024. *AToM: Amortized Text-to-Mesh using 2D Diffusion*. arXiv preprint arXiv:2402.00867.
- [7] Qiu, L., Chen, G., Gu, X., Zuo, Q., Xu, M., Wu, Y., Yuan, W., Dong, Z., Bo, L. and Han, X., 2023. *RichDreamer: A Generalizable Normal-Depth Diffusion Model for Detail Richness in Text-to-3D*. arXiv preprint arXiv:2311.16918.
- [8] Schuhmann, C., Beaumont, R., Vencu, R., Gordon, C., Wightman, R., Cherti, M., Coombes, T., Katta, A., Mullis, C., Wortsman, M. and Schramowski, P., 2022. *Laion-5b: An open large-scale dataset for training next generation image-text models*. Advances in Neural Information Processing Systems, 35, pp.25278-25294.

- [9] Shi, Y., Wang, P., Ye, J., Long, M., Li, K. and Yang, X., 2023. *Mvdream: Multi-view diffusion for 3d generation*. arXiv preprint arXiv:2308.16512.
- [10] Xie, K., Lorraine, J., Cao, T., Gao, J., Lucas, J., Torralba, A., Fidler, S. and Zeng, X., 2024. *LATTE3D: Large-scale Amortized Text-To-Enhanced3D Synthesis*. arXiv preprint arXiv:2403.15385.

Детектирование искусственно сгенерированного научного текста

Стельмах Т.Д., СПбГУ, Санкт-Петербург tanya252002@gmail.com

Аннотация

В рамках данной работы рассматривается проблема детектирования искусственно сгенерированных научных текстов, что важно для поддержания достоверности и качества научных публикаций, а также защиты академической честности. Существующие методы детектирования имеют ограничения, особенно в отношении устойчивости к перефразированию. В данной работе предлагается использование методов топологического анализа данных (TDA) для детектирования таких текстов, которые показали высокую точность и устойчивость. Был создан датасет и проведено тестирование методов на реальных и сгенерированных текстах, что позволило выявить преимущества предложенных подходов.

Введение

В последние годы технологии искусственного интеллекта, такие как популярные модели ChatGPT (для генерации текста), Sora (для генерации видео) и Suno (для генерации музыки), становятся всё более доступными и широко используемыми. Эти технологии оказывают значительное влияние на различные сферы нашей жизни, от развлечений и медиа до науки и образования.

Подробнее рассмотрим научную область. Языковые модели способны быстро и убедительно генерировать научные тексты, включая ложные факты. Кроме того, такие тексты могут искусственно повышать рейтинг цитируемости отдельных научных изданий и увеличивать количество публикаций у отдельных учёных, что ведёт кискажению научной репутации и обесцениванию научных достижений. Это также может широко использоваться в студенческих и школьных работах, что приводит к снижению качества образования. Например, студенты могут использовать искусственно сгенерированные тексты для написания курсовых и дипломных работ, не приобретая при этом реальных знаний и навыков.

В связи с этим, в рамках текущей работы мы будем рассматривать проблему детектирования искусственно сгенерированных научных текстов, в частности, статей. Это исследование важно для поддержания достоверности и ка-

чества научных публикаций, а также для защиты академической честности и предотвращения распространения дезинформации.

Существующие методы детектирования сгенерированных текстов имеют значительные ограничения, особенно в отношении устойчивости к перефразированию. Например, при задании модели контекста, в котором она должна сгенерировать текст, имитирующий человеческий, многие детекторы оказываются неэффективными. Это особенно критично в условиях, когда генеративные модели становятся всё более продвинутыми и способны создавать тексты, практически неотличимые от написанных человеком.

Для иллюстрации можно рассмотреть детектор DetectGPT. Как видно на рисунке 1, этот детектор плохо справляется с перефразированными текстами, иногда даже давая нулевую вероятность того, что текст был сгенерирован.

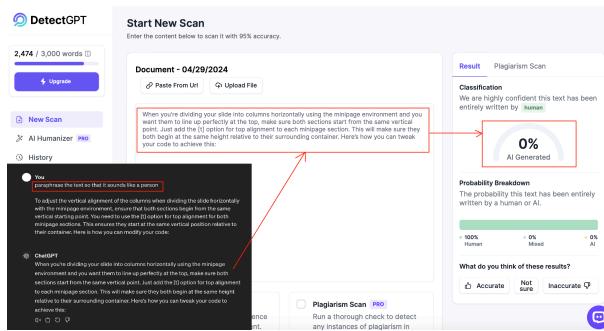


Рис. 1: Пример работы детектора DetectGPT

Таким образом, существует острая необходимость в разработке детектора, который был бы устойчив к перефразированию и изменению запросов (prompt) модели.

Обзор методов

Существует множество подходов к детектированию искусственно сгенерированных текстов. Рассмотрим основные из них:

- Статистические методы:** Статистические методы анализируют различные статистические особенности текста, такие как частоты слов (частотный анализ n-грамм), длины предложений и словарный запас [1]. Например, модели могут выявлять аномалии в распределении этих признаков, которые указывают на искусственное происхождение текста. Такие методы могут быть достаточно эффективными, но их точ-

ность может снижаться при работе с текстами, которые были специально перефразированы для обхода детекторов.

2. **Методы глубокого обучения:** Современные методы глубокого обучения, такие как BERT или GPT, могут быть дообучены для распознавания признаков искусственного текста [5]. Эти модели анализируют последовательности слов и контекстуальные связи, что позволяет им выявлять тонкие отличия между реальными и сгенерированными текстами. Методы глубокого обучения показывают высокую точность, но требуют значительных вычислительных ресурсов для обучения и работы.
3. **Измерение размерности подпространства представления данных:** Этот метод основывается на идее, что данные часто образуют многообразие меньшей размерности, чем всё пространство, в котором они представлены [6]. Мы стараемся измерить эту размерность, используя модели типа RoBERTa. Для этого необходимо взять токенизатор, пропустить через него данные, а затем подать их на вход модели для получения эмбеддингов. Далее анализируется вектор первого токена, который агрегирует информацию всего текста. Этот метод позволяет выявлять структурные особенности текста, указывающие на его искусственное происхождение.

Метод

В ходе изучения научных статей и материалов, а также проведённых самостоятельных экспериментов, было принято решение использовать методы топологического анализа данных (TDA) для детектирования искусственно сгенерированных текстов. Методы TDA показали высокую точность и устойчивость по сравнению с традиционными статистическими методами [6].

Основные преимущества методов PHD и MLE:

- **Универсальность:** Способны детектировать тексты, сгенерированные различными моделями, включая новые модели.
- **Робастность:** Более устойчивы к шуму по сравнению с другими методами, связанными с подсчётом топологических размерностей.
- **Устойчивость к перефразированию:** Метод PHD особенно хорошо справляется с перефразированными текстами.

- **Стабильность при смене темы текста:** Методы остаются эффективными независимо от тематики текста.

Остальные методы не попали в наше сравнение, поскольку либо они слишком ресурсозатратны, либо их метрики значительно хуже.

Создание датасета

Для детектирования искусственно сгенерированных научных статей потребовалось создать собственный датасет, так как подходящего готового датасета с научными статьями не нашлось.

Наиболее доступными и эффективными англоязычными языковыми моделями, согласно leaderboard'ам, оказались ChatGPT 3.5 (относительно дешевое API) и модели семейства LLaMA (бесплатные). Эти модели использовались для генерации текста, подавая на вход предложение, написанное человеком, и продолжая его. Тексты в среднем не превышали двух-трёх абзацев.

В качестве частей для детектирования были выбраны Abstract и Introduction. Abstract является кратким изложением содержания статьи, а Introduction — введением, которое часто не несет значительной смысловой нагрузки. Это делает эти части особенно подверженными генерации текста.

Темы статей были связаны с программированием, анализом данных и машинным обучением, так как специалисты в этих областях обычно лучше осведомлены о новинках ИИ. Важно было выбирать статьи, написанные до расцвета LLM (до 2018 года). Эти темы и ключевые слова для поиска статей были сгенерированы языковой моделью, и по каждой теме было найдено не менее 20 ключевых слов для поиска на Google Scholar.

Для парсинга статей был использован старый парсер 2015 года [7], который обновили и модифицировали для извлечения полного текста статьи. Поскольку многие PDF-ридеры некорректно восстанавливают текст, было принято решение использовать языковую модель для обработки полученного текста.

Результаты

Результаты измерения внутренней размерности показали, что научные тексты имеют большую внутреннюю размерность по сравнению с текстами из Википедии [6]. Это связано с более специализированной лексикой и сложными структурами предложений в научных статьях. Также диапазон внутренней

размерности для настоящих научных текстов оказался шире, что объясняется их сложностью и разнообразием тем и стиляй написания.

Также было установлено, что новые модели, такие как LLaMA 3 70B, генерируют тексты, более похожие на человеческие. Это связано с тем, что они лучше улавливают нюансы научного языка, что приводит к меньшим различиям в оцененных размерах между сгенерированными и реальными текстами. Методы MLE и PHD показали различную чувствительность к сложности текстов. PHD продемонстрировал большую чувствительность к сложным структурам научных статей, что способствовало более явному разделению между сгенерированными и реальными текстами.

Metric	PHdim SVM	PHdim GB	MLE SVM	MLE GB
Accuracy	0.59	0.59	0.60	0.59
Precision (Gen)	0.82	0.87	0.74	0.64
Precision (Human)	0.55	0.55	0.56	0.58
Recall (Gen)	0.23	0.21	0.30	0.46
Recall (Human)	0.95	0.97	0.89	0.74
F1-score (Gen)	0.36	0.34	0.43	0.54
F1-score (Human)	0.70	0.70	0.69	0.65
Validation Accuracy	0.60	0.59	0.57	0.57
ROC-AUC	0.59	0.58	0.55	0.55

Таблица 1: Результаты

В ходе сравнения метрик было замечено улучшение практически по всем важным нам показателям по сравнению с результатами, полученными с помощью датасета со статьями из Википедии [6]. Особое внимание уделялось показателю Precision (Gen), который также улучшился. Метод PHD в этом тестировании показал значительно лучшие результаты и высокую точность.

Список литературы

- [1] Gehrman, S., Strobelt, H., and Rush, A.M., 2019. *GLTR: Statistical Detection and Visualization of Generated Text*. arXiv preprint arXiv:1906.04043.
- [2] Zellers, R., Holtzman, A., Rashkin, H., Bisk, Y., Farhadi, A., Roesner, F., and Choi, Y., 2019. *Defending against Neural Fake News*. Advances in Neural

- Information Processing Systems, 32. Available at: <https://arxiv.org/abs/1905.12616>.
- [3] Kirchenbauer, J., Geiping, J., Wen, Y., Katz, J., Miers, I., and Goldstein, T., 2023. *A Watermark for Large Language Models*. arXiv preprint arXiv:2301.10226. Available at: <https://arxiv.org/abs/2301.10226>.
- [4] Krishna, K., Song, Y., Karpinska, M., Wieting, J., and Iyyer, M., 2023. *Paraphrasing Evades Detectors of AI-Generated Text, but Retrieval is an Effective Defense*. arXiv preprint arXiv:2303.13408. Available at: <https://arxiv.org/abs/2303.13408>.
- [5] Guo, B., Zhang, X., Wang, Z., Jiang, M., Nie, J., Ding, Y., Yue, J., and Wu, Y., 2023. *How Close is ChatGPT to Human Experts? Comparison Corpus, Evaluation, and Detection*. arXiv preprint arXiv:2301.07597. Available at: <https://arxiv.org/abs/2301.07597>.
- [6] Tulchinskii, E., Kuznetsov, K., Kushnareva, L., Cherniavskii, D., Barannikov, S., Piontkovskaya, I., Nikolenko, S., and Burnaev, E., 2023. *Intrinsic Dimension Estimation for Robust Detection of AI-Generated Texts*. arXiv preprint arXiv:2306.04723. Available at: <https://arxiv.org/abs/2306.04723>.
- [7] fxmlzn, 2023. *Scholar*. GitHub repository. Available at: <https://github.com/fjxmlzn/scholar>.

Вычислительная стохастика и статистические модели, многокритериальная стохастическая оптимизация



Голяндина Нина Эдуардовна

д.ф.-м.н., доцент, заведующая кафедрой статистического моделирования



Ермаков Сергей Михайлович

д.ф.-м.н., профессор, профессор кафедры статистического моделирования

Бустинговый алгоритм дисперсионного анализа на блок-схемах с медико-биологическими приложениями

Алексеева Н.П., доцент кафедры статистического моделирования СПбГУ,
 Санкт-Петербург nina.alekseeva@spbu.ru,
 Подлеснов Я.С., студент 4 курса мат-мех факультета, Санкт-Петербург
 st087301@student.spbu.ru

Аннотация

В статье изучается проверка значимости факторов линейной модели через бустинговый алгоритм, то есть путем много-кратного применения модели дисперсионного анализа на блок-схемах для несбалансированных ретроспективных данных с последующим изучением распределения полученных статистик. Определена вариантность блок-схемы для заданной таблицы сопряженности факторов с учетом возможной неполноты последней. Комбинаторным образом получено число возможных вариантов для полных блок-схем и для типа Q. Построено два алгоритма перечисления вариантов, по которым строятся статистик проверки значимости влияния сбалансированных факторов.

Введение

Блок-схема или дизайн $D(v, b, r, k, \lambda)$ определяется как таблица, состоящая из v элементов, сгруппированных по b блокам размера k так, что каждый элемент встречается r раз, а каждая пара встречается λ раз. Если число блоков меньше C_v^k , то такая схема называется неполной сбалансированной. Известны два необходимых соотношения баланса $vr = bk$, $\lambda(v - 1) = r(k - 1)$, актуальные для идентификации блок-схем. В комбинаторике [1] большое внимание уделено теоремам существования отдельных блок-схем и методам их построения. В планировании эксперимента с помощью блок-схем удается существенно снизить число наблюдений при изучении влияния нескольких факторов типа методов обработки на некоторую зависимую переменную. Для того чтобы уменьшить число наблюдений, разные методы обработки при различных условиях применяются по специальной схеме. Дисперсионному анализу на блок-схемах посвящена классическая монография [2].

Идея данной работы состоит в том, чтобы применить этот метод на ретроспективных данных с большим числом градаций. Для этого предполагается проводить дисперсионный анализ на частичных подвыборках, градации

факторов которых образуют блок-схему, а вывод о значимости факторов осуществлять по ансамблю полученных статистик. Этим объясняется термин бустинговость, поскольку общий вывод осуществляется на основе анализа повторяемости заведомо более слабых выводов по фрагментам выборки. Задача заключается, с одной стороны, выборе подходящей блок-схемы, с другой стороны, в оценке числа вариантов ее применения. Ответы на эти вопросы в большей степени определяются типом выбранной блок-схемы и ее комбинаторными свойствами.

Об отборе сбалансированных данных

В табл. 1 представлены данные о среднем проценте пораженных легких у больных ковидом в зависимости от возрастной группы и порогового уровня белка острой фазы¹. Для краткости столбцы, которые отражают возрастную группу, пронумерованы от 0 до 5. Строки таблицы также перенумерованы от нуля до девяти, а в последней строке указаны номера подчеркнутых строк. В каждой строке, отвечающей некоторому уровню белка СРБ, подчеркнуты какие-то три значения. Если рассмотреть номера столбцов с подчеркнутыми значениями (столбец b_j), то можно непосредственно убедиться в том, что номера шести столбцов, сгруппированные по три в 10 блоков, образуют блок-схему $D(6, 10, 5, 3, 2)$. В качестве геометрической иллюстрации дизайна $D(6, 10, 5, 3, 2)$ можно рассматривать пятиугольник, вершины которого соединены с центром. Пять внутренних треугольников образуют пять блоков. Остальные пять блоков представляют собой треугольники со стороной в виде ребра пятиугольника, соединенного с противоположной вершиной (рис. 1).

Наблюдения с выделенными сочетаниями факторов образуют сбалансированную частичную подвыборку, для которой может быть применен дисперсионный анализ на блок-схемах.

Дисперсионный анализ на блок-схемах

Модель дисперсионного анализа на блок-схемах применяется для сбалансированной выборки и имеет вид

$$x_{ij} = \mu + v_i + b_j + \varepsilon_{ij}, \quad \varepsilon_{ij} \sim \mathcal{N}(0, \sigma),$$

где через v_i и b_j обозначены дифференциальные эффекты факторов, указанных соответственно в строках и столбцах таблицы 1. Основные статистики –

¹Белок острой фазы или СРБ возрастает в организме при наличии воспалительного процесса

возраст	[22,44]	0	(44,55]	(55,65]	(65,76]	(76,87]	(87,98]	b_j
CRB	0	1	2	3	4	5		
(0,2.5]	0	<u>19</u>	<u>21</u>	25	36	47	<u>10</u>	015
(2.5,2.85]	1	16	<u>10</u>	<u>28</u>			<u>10</u>	125
(2.85,3.2]	2	<u>20</u>		<u>30</u>	38	<u>60</u>		024
(3.2,3.55]	3	45	28	<u>40</u>	<u>31</u>	29	<u>28</u>	235
(3.55,3.9]	4	58	88	60	<u>53</u>	<u>36</u>	<u>16</u>	345
(3.9,4.25]	5	50	<u>48</u>	61	<u>68</u>	<u>36</u>		134
(4.25,4.6]	6	<u>43</u>	43	60	71	<u>59</u>	<u>65</u>	045
(4.6,4.95]	7	<u>85</u>	64	<u>77</u>	<u>72</u>	56	<u>55</u>	023
(4.95,5.3]	8	<u>15</u>	<u>69</u>	68	<u>70</u>	76	39	013
(5.3,5.65]	9		<u>66</u>	<u>76</u>	35	<u>25</u>		124
γ_i	02678	01589	12379	34578	24569	01346		

Таблица 1: Средний объем поражения легких в зависимости от уровня белка CRB и от возраста больных ковидом.

это суммы по строкам $V_i = \sum_j x_{ij}$ и по столбцам $B_j = \sum_i x_{ij}$, а также сумма $T_i = \sum_{j \in \gamma_i} B_j$. Оценки параметров по методу наименьших квадратов [2] имеют вид

$$\hat{\mu} = \frac{1}{bk} \sum_{i,j} x_{ij}, \quad \hat{v}_i = \frac{kV_i - T_i}{\lambda v}, \quad \hat{b}_j = \frac{1}{k} (B_j - \sum_{i \in b_j} \hat{v}_i) - k\hat{\mu},$$

где суммирование (i, j) означает или $\sum_{i=1}^v \sum_{j \in \gamma_i}$, или $\sum_{j=1}^b \sum_{i \in \beta_j}$, а b_j и γ_i соответственно блоки прямого (b_j) и двойственного (γ_i) дизайна² (табл.1). Статистические критерии опираются на разложение суммы квадратов отклонений

²Двойственный дизайн образуют блоки прямого дизайна, взятые в качестве элементов, агрегированные по инцидентности отдельного элемента прямого дизайна.

$$\sum_{i,j} (x_{ij} - \hat{\mu})^2 = S_e^2 + S_v^2 + S_b^2, \text{ где}$$

$$S_v^2 = \sum_{ij} \left(\hat{v}_i - \frac{1}{k} \sum_{i \in \beta_j} \hat{v}_i \right)^2, \quad S_b^2 = \sum_{j=1}^b k \left(\frac{B_j}{k} - \hat{\mu}^2 \right)^2,$$

$$S_e^2 = \sum_{i,j} \left(x_{ij} - \hat{v}_i + \frac{1}{k} \sum_{l \in \beta_j} \hat{v}_i - \frac{B_j}{k} \right)^2.$$

При справедливости гипотез о незначимости факторов статистики

$$F = \frac{S_v^2}{S_e^2} \cdot \frac{bk - v - b + 1}{v - 1} \quad \text{и} \quad F = \frac{S_b^2}{S_e^2} \cdot \frac{bk - v - b + 1}{b - 1}$$

имеют распределение Фишера соответственно с $df_v = v - 1$, $df_e = bk - v - b + 1$ и с $df_b = b - 1$, $df_e = bk - v - b + 1$ степенями свободы.

Комбинаторный аспект вариантности дизайна

Очевидно в таблицу с b строками и v столбцами можно вписать очевидно дизайн с v элементами и b блоками. Из табл. 1 имеем $v = 6$ на $b = 10$. Известен дизайн $D(6, 10, 5, 3, 2)$ с таким соотношением числа элементов и числа блоков [1]. Он является остаточным для блок-схемы $D(11, 5, 2)$ типа Q^3 . Группой автоморфизмов⁴ дизайна $D(6, 10, 5, 3, 2)$ является группа $PSL_2^{F_5}$ симметрий додекаэдра.⁵ Элементам этой группы соответствуют способы расположения вершин графа (рис.1) с сохранением блоков. Например, в центр можно поставить любой из шести элементов, пусть 5, вершину 0 можно обозначить оставшимися пятью способами, а для обозначения вершины 1 остаются только два варианта: либо 1, либо 4, так как пара 05 входит в состав двух блоков 015 и 045. Для оставшихся вершин все однозначно. Таким образом получаем $60 = 6 \cdot 5 \cdot 2$ автоморфизмов.

³К типу Q относятся блок схемы, у которых орбита представляет собой квадратичные вычеты по полю F_q . Например, для $q = 11$ квадратичными вычетами являются числа 1, 3, 4, 5, 9. Если построить одиннадцать блоков вида $(1 + j, 3 + j, 4 + j, 5 + j, 9 + j)$, где $j = 0, 1, \dots, 10$, то получим дизайн $D(11, 11, 5, 5, 2)$, остаточным к которому оказывается $D(6, 10, 5, 3, 2)$.

⁴Подстановка на элементах блок-схемы D , относительно которой инвариантно множество блоков, называется автоморфизмом дизайна.

⁵В $PSL_2^{F_5}$ буква L означает, что это линейная группа матриц, индекс 2 — порядка два над полем F_5 с единичным определителем (S). В проективной группе (P) матрицы, отличающиесяомножителем, задают одно и то же преобразование.

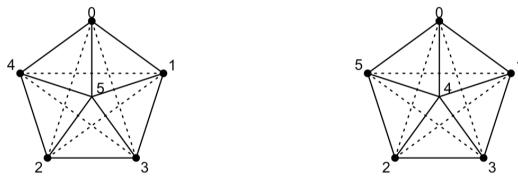


Рис. 1: Геометрическая интерпретация неавтоморфных подстановок дизайна $D(6, 10, 5, 3, 2)$.

Неавтоморфными будем называть подстановки с неодинаковыми множествами блоков[3]. Для того чтобы вычислить S_D число неавтоморфных подстановок дизайна, нужно $v!$ подстановок на элементах разделить на число его автоморфизмов. Например, дизайн $D(6, 10, 5, 3, 2)$ имеет $S_D = \frac{6!}{60} = 12$ неавтоморфных подстановок. Очевидно половина подстановок являются противоположными, а шесть неавтоморфных подстановок можно получить, если зафиксировать, к примеру, тройку элементов 0, 1, 5, а на оставшихся трех 2, 3, 4 задать шесть подстановок (рис. 1).

Итак, если в матрице наблюдений типа табл.1 выбрать b строк и v столбцов так, что r элементов встречаются в каждом столбце и k элементов в каждой строке, а каждая пара встречается λ раз, то мы получаем сбалансированную решетку в виде некоторой подстановки дизайна $D(v, b, r, k, \lambda)$ с упорядоченными блоками. Совокупность всех вариантов вложения дизайна в таблицу (учитываются только ненулевые ячейки), будем называть вариантностью. Полный порядок вариантности определяется числом неавтоморфных подстановок, умноженных на число перестановок блоков, то есть $b! \cdot S_D$.

Алгоритмический аспект вариантности дизайна

Поскольку в реальности матрица наблюдений редко представлена полным набором всевозможных сочетаний градаций факторов, число решеток оказывается намного меньше $b! \cdot S_D$. Например, во второй строке таблицы 1 представлены только элементы 0, 1, 2, 5. Поэтому во второй строке могут быть только два блока $B_0 = 015$ и $B_1 = 125$. Аналогично в третьей строке с элементами 0, 2, 3, 4 могут располагаться только блоки $B_2 = 024$ и $B_7 = 023$, а в десятой строке с элементами 1, 2, 3, 4 могут располагаться только блоки $B_5 = 134$ и $B_9 = 124$. Еще в шестой строке отсутствует элемент 5, по-

этому там могут располагаться пять блоков, которые этого числа не содержат, 024, 134, 023, 013 и 124. Заметим, что в трех строках с пропущенными двумя элементами могут располагаться разные блоки, поэтому рассматривая восемь всевозможных их сочетаний, увидим, что обязательно два из них не содержат элемент 5, следовательно, получаем число перестановок блоков в этом одном из 12 дизайнов в виде $2^3 \cdot 3 \cdot 6! = 24 \cdot 6! = 17280$. Аналогично считаются перестановки блоков для остальных одиннадцати подстановок. Всего для такой структуры распределения по градациям получено $(4 \cdot 24 + 2 \cdot 20 + 4 \cdot 15 + 2 \cdot 18) \cdot 6! = 232 \cdot 6! = 167040$, что более чем в 250 раз меньше полного числа перестановок. В целом, процесс алгоритмизуется, и можно непосредственно убедиться в правильности подсчета числа необходимых решеток.

Для контроля был реализован еще один алгоритм, основанный на расстановке всевозможных вариантов блоков. На первых двух этапах все просто - выбираем первый B_1 блок $20 = C_6^3$ способами, второй B_2 восемнадцатью, поскольку исключаются первый блок и ему противоположный. Далее сложнее, поскольку из 16-ти вакантных для B_3 блоков, которые определяются парой B_1, B_2 , не все тройки блоков возможны. Но можно построить алгоритм, по которому на n -м шаге проверяется, в какие подстановки дизайна входят блоки B_1, \dots, B_n , $0 < n \leq b - 1$. Если такой дизайн один, то процедура заканчивается приведением всех оставшихся перестановок $\mathcal{P}(B_{n+1}, \dots, B_b)$. Иначе рассматриваются возможные кандидатуры на B_{n+1} . Для этого из множества элементов Ω_v дизайна исключаются предыдущие блоки B_1, \dots, B_n и им противоположные. Если представлены не все ячейки таблицы сопряженности по градациям факторов, то нужно рассматривать пересечение с множеством возможных блоков, вызванном неполнотой данных. Реализованный алгоритм в R алгоритм привел все $S_D \cdot b!$ необходимых подстановок, но с большим преимуществом во временных затратах.

Практические примеры

Для таблицы сопряженности факторов 1, в которой $b = 10$ строк и $v = 6$ столбцов, в случае полного перебора необходимо перебрать 43545600 решеток дизайна $D(6, 10, 5, 3, 2)$. Если заранее учитывать незаполненность некоторых ячеек, число вариантов сводится, как было указано ранее, к 167040. Для каждого варианта вычислены статистики Фишера и оценены значимости влияния факторов. Установлена высокая значимость влияния белка СРБ на процент поражения легких, в 93.36% p -значение не превышает уровень значимости $\alpha = 0.1$, в 80% уровень значимости $\alpha = 0.05$, в то время как фактор

возраста в 95% случаев оказался незначимым.

Второй пример связан с исследованием влияния разных методов лечения ожогов у лабораторных крыс. В качестве зависимой переменной рассматривалась площадь ожога, а в качестве факторов рассматривались способ лечения с 4-мя градациями и масса тела как показатель состояния всего организма с 6-ю градациями. Для отбора частичных выборок использовали дизайн $D(4, 6, 3, 2, 1)$ – полную блок-схему с единственной подстановкой. Были проанализированы распределения значимостей соответствующих статистик. В результате удалось выделить два существенных временных момента: 10 и 19 дней. Через 10 дней проявилось наибольшее различие между методами лечения, а через 19 дней проявилась наибольшая дифференциация по массе.

Заключение

Таким образом, сопоставляя таблицу сопряженности факторов с комбинаторными свойствами адекватных блок-схем, можно заранее оценить число возможных статистических итераций бустингового алгоритма. Показана целесообразность изучения комбинаторных свойств дизайнов и двойственный подход в параметризации вариантности дизайнов. В дальнейшем предполагается расширить алгоритм на блок-схемы других типов и изучить свойства нового критерия на модельных данных.

Список литературы

- [1] М. Холл. Комбинаторика. — Изд-во «МИР», Москва, 1970.
- [2] Д. Дюге. Теоретическая и прикладная статистика. — Главная редакция физико-математической литературы изд-ва «Наука», 1972.
- [3] Н.П. Алексеева. Анализ медико-биологических систем. Реципрокность, эргодичность, синонимия. — Изд-во С.Петербург. ун-та, 2012.

Подход к сравнению методов обнаружения разладки

Шаповал Е.А., СПбГУ, Санкт-Петербург st063753@student.spbu.ru,
Голяндина Н.Э., СПбГУ, Санкт-Петербург n.golyandina@spbu.ru¹

Аннотация

Рассматривается задача обнаружения разладки во временных рядах с помощью функции разладки. Для сравнения методов обнаружения разладки предлагается подход, аналогичный использованию ROC кривых в задачах классификации. Преимуществом подхода является то, что для сравнения методов на требуется задавать порог, при превышении которого метод сигнализирует о разладке. Особенности подхода продемонстрированы на примере. Показано, что кривые, параметризованные порогом, где по оси X откладывается величина, обратная к средней длине интервала без сигнала о разладке в случае ее отсутствия, а по оси Y — вероятность правильного обнаружения разладки с опозданием не больше заданного, являются предпочтительными для сравнения.

Введение

В работе рассматривается задача обнаружения разладки в сигнале при наблюдаемом зашумленном сигнале: $X_N = S_N + R_N$, где X_N — наблюдаемый временной ряд длины N , $S_N \in \mathbb{R}^N$ — сигнал, $R_N \in \mathbb{R}^N$ — шум, т.е. стационарный случайный процесс.

Разладку можно определить, как некоторое изменение в структуре ряда. Приведём самый простой пример. Пусть S_N — сигнал с общим членом

$$s_n = \begin{cases} 0, & n = 1, \dots, Q - 1, \\ \mu, & n = Q, \dots, N, \end{cases} \quad (1)$$

$2 < Q < N$, $\mu > 0$, и R_N — шум с общим членом $r_n \sim \mathcal{N}(0, \sigma^2)$, $\sigma > 0$. Тогда временной ряд $X_N = S_N + R_N$ можно рассмотреть как ряд, имеющий разладку в среднем случайного процесса.

Более сложный пример — это разладка в частоте сигнала:

$$s_n = \begin{cases} A \cos(2\pi\omega_1 n), & n = 1, \dots, Q - 1, \\ A \cos(2\pi\omega_2 n + \varphi), & n = Q, \dots, N, \end{cases} \quad (2)$$

¹Работа поддержана грантом РНФ 23-21-00222.

$$2 < Q < N, A > 0, |\omega_1 - \omega_2| > \Delta > 0.$$

Далее точку Q будем называть *точкой разладки*.

Для обнаружения разладки существует много разных методов, зависящих, в частности, от структуры сигнала и вида разладки [1, 2]. Целью работы является построение способа сравнения методов обнаружения разладки, аналогичного использованию ROC-кривых в задачах классификации [3].

Обнаружение разладки

Функция разладки

Как правило, для обнаружения разладки строится функция разладки.

Определение 1. Рассмотрим временной ряд X_N , зафиксируем параметр $L = 2, \dots, N - 1$ — *длина окна*. *Функцией разладки* назовём отображение $h : \{L, L + 1, \dots, N\} \rightarrow \mathbb{R}_+$, у которого каждое значение h_{i+L} зависит от отрезка временного ряда $(x_{i+1}, \dots, x_{i+L})$, $i = 0, \dots, N - L$. Назовём такой отрезок *тестовым* и заметим, что функция разладки синхронизирована с его концом.

В качестве примера функции разладки для обнаружения разладки в среднем, если до этого среднее было равно нулю, рассмотрим

$$h_{i+L} = \sum_{j=i+1}^{i+L} x_j^2, \quad (3)$$

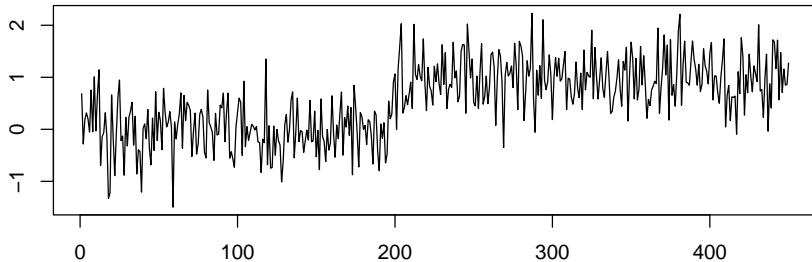
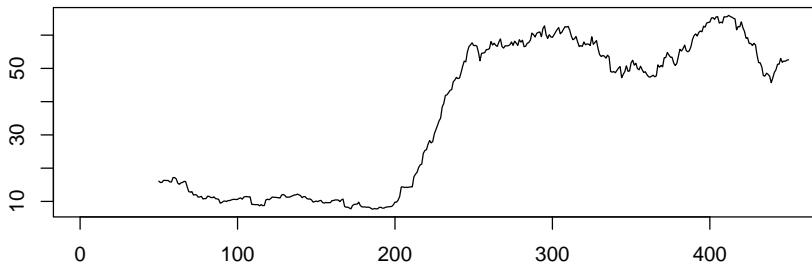
$$i = 0, \dots, N - L.$$

На рис. 1 изображён временной ряд с разладкой в среднем случайного процесса с сигналом вида (1), где параметр $\mu = 1$, а значение точки разладки $Q = 200$. На рис. 2 изображена его функция разладки вида (3). Заметим, что после точки разладки Q функция разладки h начинает возрастать. Таким образом, можно её использовать как индикатор разладки во временном ряде.

Оценка точки разладки

Введем некоторое число $\theta > 0$, далее будем называть это число *порогом*. Пусть h — функция разладки ряда X_N , оценим точку разладки Q как

$$\hat{Q}(h, \theta) := \min\{i \in \{L, L + 1, \dots, N\} : h_i > \theta\}.$$

Рис. 1: Временной ряд X_N , $\mu = 1$ Рис. 2: Функция разладки ряда X_N , $\mu = 1$

Выделим две задачи, которые нужно решить для определения точки разладки указанным способом: выбор вида функции разладки и выбор значения порога θ .

Для оценки качества работы алгоритма по обнаружению точки разладки зафиксируем натуральное k , назовём его *допустимым временем запаздывания*. Пусть \hat{Q} — оценка Q . Определим следующие виды срабатываний:

- $\hat{Q} \in [Q, Q + k]$ — оценка Q правильная, правильное срабатывание (true alarm),
- $\hat{Q} < Q$ — ложное срабатывание (false alarm),
- $\hat{Q} > Q + k$ — срабатывание с запаздыванием (delay).

Для оценки false alarm и true alarm выделим два подхода: через вероятности и «в среднем». Для оценки вероятности false alarm будем использовать

$$\text{FPR} := P(\hat{Q} < Q),$$

обратим внимание, что FPR зависит от значения Q .

Оценка вероятности true alarm может быть записана как

$$\text{TPR} := \mathbb{P}(\hat{Q} \in [Q, Q + k] \mid \hat{Q} \geq Q).$$

Учитывая что $\text{FPR} = \text{FPR}(Q, h, \theta)$ и $\text{TPR} = \text{TPR}(k, h, \theta)$ — монотонно невозрастающие функции по θ , можно выделить две постановки задачи.

- Максимизировать TPR при условии $\text{FPR} < \alpha$, где $\alpha \in (0, 1)$ — максимально допустимая вероятность false alarm за время Q ; эта постановка задачи стандартная и в ней легче получать теоретические результаты.
- Минимизировать FPR при условии $\text{TPR} > \gamma$, где $\gamma \in (0, 1)$ — вероятность своевременного обнаружения разладки; такая постановка задачи ближе к практике, но в ней нужно конкретизировать вид разладки.

В подходе «в среднем» вычисляют

$$\text{FARL} := \mathbb{E}\hat{Q}$$

для ряда без разладки, а для ряда с разладкой —

$$\text{TARL} := \mathbb{E}(\hat{Q} \mid \hat{Q} \geq Q).$$

Аналогично подходам через вероятности можно сформулировать задачи, используя FARL и/или TARL. Выбор порога в обеих постановках — сложная задача.

Примеры постановок задач

Опишем пример первой постановки задачи, с условием $\text{FPR} < \alpha$. Будем рассматривать временной ряд с сигналом вида (1), для функции разладки вида (3) аппроксимацию значений FPR и FARL для больших Q можно найти аналитически [4]. Затем по найденным аппроксимациям можно находить порог θ . Данный пример можно обобщить до следующего вида. Общий член сигнала S_N имеет вид

$$s_n = \begin{cases} u_n, & n = 1, \dots, Q - 1, \\ u_n + \mu, & n = Q, \dots, N, \end{cases}$$

$\mu > 0$, временной ряд $U_N = (u_1, u_2, \dots, u_N)$ выделяется некоторым методом, например, методом анализа сингулярного спектра (SSA) [5], а далее разработанная теория применяется к остатку.

Пример второй постановки задачи рассмотрен в [6], для сигнала вида (2), функция разладки строилась на основе метода анализа сингулярного спектра (SSA). В работе был теоретически найден порог, при котором в наихудшем по ω_2 и φ случае выполняется неравенство $\text{TPR} > \gamma$.

Способ сравнения методов обнаружения разладки

Рассмотрим два вида аналога ROC-кривых для определения качества метода обнаружения разладки.

Определение 2. Рассмотрим функцию разладки h временного ряда X_N и набор порогов $\Theta \subset \mathbb{R}$. Назовем FPR – TPR *кривой* параметрически заданную кривую

$$\{(x, y) \in \mathbb{R}^2 : x = \text{FPR}(Q, h, \theta), y = \text{TPR}(k, h, \theta), \theta \in \Theta\}$$

и FARL – TPR *кривой* параметрически заданную кривую

$$\{(x, y) \in \mathbb{R}^2 : x = 1/\text{FARL}(h, \theta), y = \text{TPR}(k, h, \theta), \theta \in \Theta\}.$$

Обе кривые определены так, что чем выше лежит кривая, тем лучше работает метод обнаружения разладки. Введённые кривые позволяют сравнивать методы для всего множества порогов, без вычисления конкретного.

Для демонстрации использования введённых кривых рассмотрим пример с разладкой в среднем и введём два вида функции разладки. Сначала определим функцию разладки с весами.

Рассмотрим временной ряд X_N , зафиксируем параметр L , рассмотрим вектор весов $w = (w_1, \dots, w_L) \in \mathbb{R}^L$. Функцией разладки с весами w назовём функцию

$$h_{i+L} := \sum_{j=1}^L w_j x_{i+j}^2,$$

$$i = 0, \dots, N - L.$$

Рассмотрим функции разладки с двумя видами весов:

- стандартная (unweighted), $h_{i+L}^0 := \sum_{j=1}^L x_{i+j}^2$, $w^0 = (1, 1, \dots, 1)$,
- взвешенная (weighted), $h_{i+L}^{(k)} := \sum_{j=k}^L x_{i+j}^2$, $w^{(k)} = (0, \dots, 0, \underbrace{1, \dots, 1}_k)$.

Численные эксперименты

В качестве примера будем рассматривать временной ряд с разладкой в среднем, то есть сигнал вида (1), параметры $Q = 60$ и $Q = 150$, $\sigma = 1$, $L = 50$, $k = 10$. Оценки FPR, TPR и FARL будем проводить по выборке объёма $n = 1000$.

Сначала построим графики для случая без разладки, $\mu = 0$, для разных значений Q для того, чтобы относительно него оценивать эффективность метода обнаружения разладки. На рис. 3 видно, что для различных Q получаем различные исходные оценки даже при отсутствии разладки. Более того, получаем разную упорядоченность методов даже при отсутствии разладки.

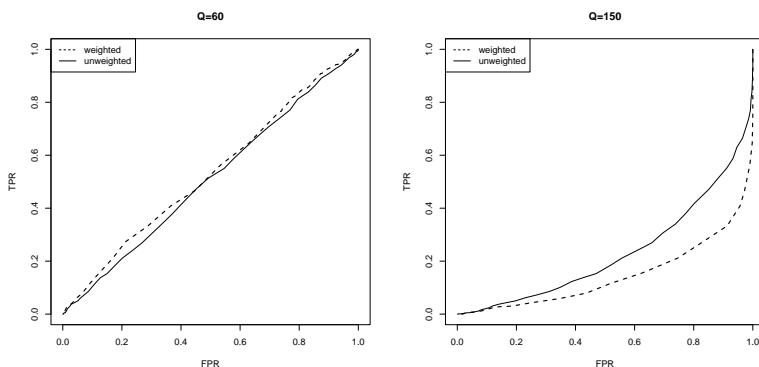


Рис. 3: FPR – TPR кривые, $\mu = 0$, $Q = 60$ (слева) и $Q = 150$ (справа)

Теперь рассмотрим случай с маленькой разладкой ($\mu = 0.5$). На рис. 4 видно, что для $Q = 60$ взвешенный вариант функции разладки лучше, а для $Q = 150$ лучше стандартный; то есть, результат сравнения методов зависит от значения Q .

Для того чтобы результат сравнения методов не зависел от Q , вместо значения FPR по оси X будем откладывать значение $1/\text{FARL}$, т.е. рассмотрим FARL – TPR кривые. Построим их для двух примеров с разной величиной разладки, $\mu = 0.5$ и $\mu = 2$. На рис. 5 изображены пары FARL – TPR кривых. По FARL – TPR кривым можно установить, что для примера с $\mu = 0.5$ стандартный вариант функции разладки лучше взвешенного, а для примера с большим $\mu = 2$ лучше взвешенный вариант.

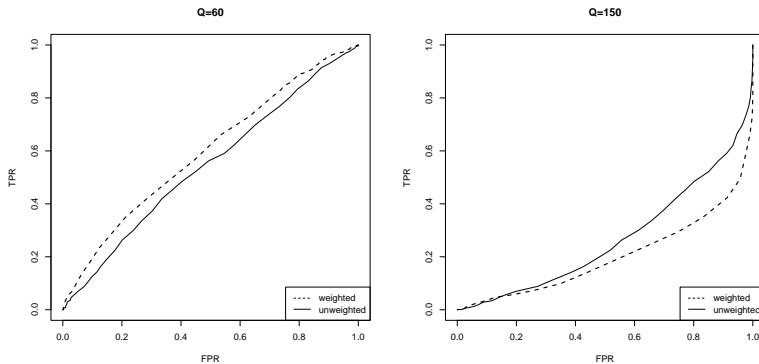


Рис. 4: FPR – TPR кривые, $\mu = 0.5$, $Q = 60$ (слева) и $Q = 150$ (справа)

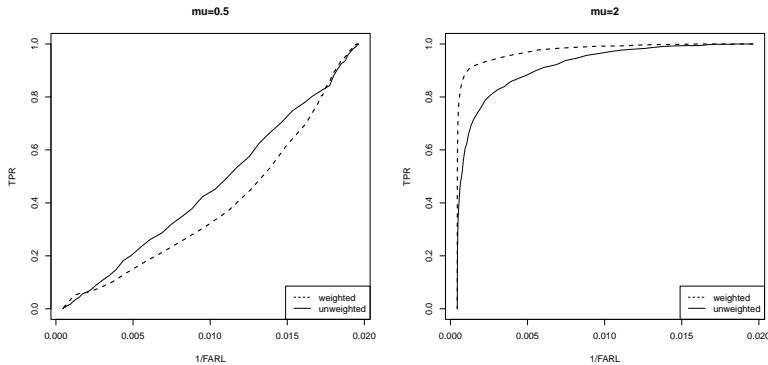


Рис. 5: FARL – TPR кривые, $Q = 60$, $\mu = 0.5$ (слева) и $\mu = 2$ (справа)

Заключение

Для сравнения способов обнаружения разладки был разработан подход, аналогичный использованию ROC кривых в задачах классификации. Преимуществом подхода является то, что для сравнения методов не требуется выполнять сложную и часто не полностью теоретически обоснованную процедуру определения подходящего порога. Особенности подхода были продемонстрированы на примере разладки в среднем гауссовского процесса. В результате, были показаны недостатки FPR – TPR кривых, вызванные тем, что вероятность FPR зависит от длины промежутка, на котором измеряется false alarm. В свою очередь, FARL – TPR кривые, где вероятность FPR заме-

няется на величину, обратную к средней длине интервала без false alarm, не обладает таким недостатком, поэтому сравнение методов обнаружения разладки по FARL – TPR кривым представляется более предпочтительным.

Список литературы

- [1] Lai, T. L. Sequential Changepoint Detection in Quality-Control and Dynamical Systems // Journal of Royal Statistical Society - Series B. — 1995 — Vol. 57, No. 4, pp. 613–658.
- [2] Moskvina, V., and Zhigljavsky, A. An Algorithm Based on Singular Spectrum Analysis for Change-Point Detection // Communications in Statistics - Simulation and Computation. — 2003 — Vol. 32, No. 2, pp. 319–352.
- [3] Bradley, A.P. The use of the area under the ROC curve in the evaluation of machine learning algorithms // Pattern recognition. — 1997 — Vol. 30. No.7, pp.1145-1159.
- [4] Noonan, J., Zhigljavsky, A. Approximations for the Boundary Crossing Probabilities of Moving Sums of Random Variables // Methodol Comput Appl Probab. — 2021 — Vol. 23, pp. 873–892.
- [5] Golyandina N., Nekrutkin V., Zhigljavsky A. Analysis of time series structure: SSA and related techniques. — 2001 — Chapman & Hall/CRC.
- [6] Шаповал Е.А. Обнаружение разладки в частоте колебания методом анализа сингулярного спектра // Выпускная квалификационная работа (научный руководитель Голяндина Н.Э.) — 2023 — СПбГУ.

О редукции параметров в моделях сложных распределений с применением в радиобиологии

Олейник М. В., СПбГУ, Санкт-Петербург st087252@student.spbu.ru,
Алексеева Н.П., СПбГУ, Санкт-Петербург nina.alekseeva@spbu.ru

Аннотация

В данной статье рассматривается возможность применения двухпараметрической модели сложного логарифмически-пуассоновского распределения для описания динамики роста числа аномалий на ядрах клеток рабдомиосаркомы в зависимости от облучения. Приведены вероятности распределения с применением чисел Стирлинга второго рода и Эйлера. Приведено обоснование возникновения этих специальных чисел через ряды по упорядоченным разбиениям.

Введение

Для радиобиологических данных числа аномалий в ядрах клеток рабдомиосаркомы у крыс при облучении разной степени интенсивности в экспериментах *in vivo* и *in vitro* ранее была исследована модель реинтрантного бинома с двумя парами неизвестных параметров. В поисках адекватных моделей с меньшим числом параметров рассматривались разные варианты, сначала обобщение отрицательного биномиального распределения за счет перехода от пуассоновского распределения числа слагаемых к биномиальному — биномиально-логарифмического, затем его антипод — логарифмически-биномиальное распределение. Предварительная оценка параметров указала на преимущество последнего в предельном случае, соответствующем пуассоновскому распределению случайных компонент. В данной статье изучаются модели пуассон-логарифмического (отрицательно-биномиальное) и логарифмически-пуассоновское (ЛПР). Эти двухпараметрические модели показали удовлетворительное согласование с эмпирическими данными.

Наиболее адекватной для интерпретации оказалась модель ЛПР. Соотношение между оценками параметров логарифмического и пуассоновского распределений свидетельствуют об интенсивном характере образования аномалий в эксперименте *in vivo* и экстенсивном в эксперименте *in vitro*.

Логарифмически-пуассоновское распределение

Логарифмически-пуассоновское распределение вводится как распределение случайной суммы независимых случайных величин

$$\zeta_\tau = \xi_1 + \cdots + \xi_\tau,$$

где ξ_i имеют распределение Пуассона с параметром λ , и производящей функцией $g(t) = e^{\lambda(t-1)}$, а τ — логарифмическое с вероятностями

$$P(\tau = j) = \frac{\alpha q^j}{j}, \text{ где } \alpha = -(\ln(1-q))^{-1}, \quad j = 1, 2, \dots$$

с производящей функцией $f(t) = -\alpha \ln(1 - qt)$. Согласно [2], производящая функция случайной величины ζ имеет вид суперпозиции производящих функций логарифмического и пуассоновского распределений,

$$h(t) = -\alpha \ln(1 - qe^{\lambda(t-1)}). \quad (1)$$

Теорема 1. Обозначим через $S(k, j)$ числа Стирлинга второго рода [3], имеющие следующую рекуррентную формулу:

$$S(k, j) = S(k-1, j-1) + j \cdot S(k-1, j), \quad 0 < j \leq k,$$

$S(0, 0) = 1, S(k, 0) = 0$ при $k > 0, S(k, j) = 0$ при $j > k$. Пусть $Q(\lambda, q) = \frac{qe^{-\lambda}}{1-qe^{-\lambda}}$, где $\lambda > 0$ и $q \in (0, 1)$. Тогда вероятности логарифмически-пуассоновского распределения с параметрами λ, q и с производящей функцией (1) имеют вид:

$$P(\zeta_\tau = k) = \frac{\alpha}{k!} \lambda^k \sum_{j=1}^k (j-1)! S(k, j) Q(\lambda, q)^j \text{ при } k = 1, 2, \dots, \quad (2)$$

$$P(\zeta_\tau = 0) = -\alpha \ln(1 - qe^{-\lambda}).$$

Вероятности (2) получены через формулу для производящих функций: $P(S_\tau = k) = \frac{1}{k!} h^{(k)}(0)$, $k = 0, 1, 2, \dots$, а доказательство теоремы осуществлено по методу математической индукции. Для чисел Стирлинга второго рода известна явная формула $S(k, j) = \frac{1}{j!} \sum_{i=0}^j (-1)^{j+i} C_j^i i^k$, поэтому не видно никаких ограничений для использования (2). Однако вероятности (2) можно получить при помощи специальных чисел другого вида. Также методом математической индукции доказано следующее утверждение.

Теорема 2. Обозначим через $E(k, j)$ числа Эйлера первого рода [3] для которых справедлива рекуррентная формула:

$$E(k, j) = (j + 1) \cdot E(k - 1, j) + (k - j) \cdot E(k - 1, j - 1), \quad 0 < j < k - 1,$$

$$E(k, 0) = 1 \text{ при } k \geq 0, \quad E(k, j) = 0 \text{ при } j \geq k > 0.$$

Вероятности логарифмически-пуассоновского распределения для $k > 0$ могут быть выражены следующим образом:

$$P(\zeta_\tau = k) = \frac{\alpha}{k!} \frac{\lambda^k q e^{-\lambda}}{(1 - q e^{-\lambda})^k} \sum_{j=0}^{k-2} E(k - 1, j) (q e^{-\lambda})^j. \quad (3)$$

Числа Эйлера (табл.1) представляют собой количество перестановок порядка k с j подъёмами, то есть в перестановке $\pi = (\pi_1, \dots, \pi_k)$ всего j индексов i таких, что $\pi_i < \pi_{i+1}$.

$k \setminus j$	0	1	2	3	4	5
0	1					
1	1					
2	1	1				
3	1	4	1			
4	1	11	11	1		
5	1	26	66	26	1	
6	1	57	302	302	57	1

Таблица 1: Числа Эйлера.

Эти числа, явный вид которых $E(k, j) = \sum_{i=0}^j C_{k+1}^i (-1)^i (j+1-i)^k$, возникают также в функциональном ряде

$$\sum_{i=1}^{\infty} i^k x^i = \frac{x}{(1-x)^{k+1}} \sum_{j=0}^{k-1} E(k, j) x^j,$$

который в нашем случае возникает при использовании формулы полной вероятности

$$\begin{aligned} P(\zeta_\tau = k) &= \sum_{j=1}^{\infty} P(\zeta_\tau = k | \tau = j) \cdot P(\tau = j) = \\ &= \sum_{j=1}^{\infty} -\frac{1}{\ln(1-q)} \frac{q^j}{j} \frac{(j\lambda)^k}{k!} e^{-j\lambda} = \frac{\alpha}{k!} \lambda^k \sum_{j=1}^{\infty} j^{k-1} (q e^{-\lambda})^j. \end{aligned}$$

Оценка параметров и пример из радиобиологии

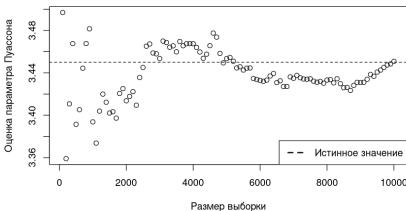


Рис. 1: Сходимость оценки параметра λ к истинному значению для логарифмически-пуассоновского распределения.

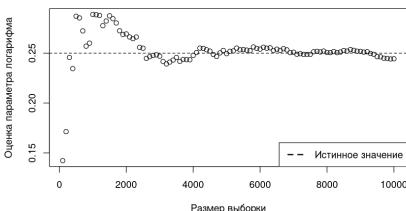


Рис. 2: Сходимость оценки параметра q к истинному значению для логарифмически-пуассоновского распределения.

Оценка параметров λ и q проводилась методом максимального правдоподобия численно на компьютере с использованием функции `optim` на языке R. Моделируя выборку из n индивидов для известных параметров можно проверить состоятельность такой оценки (рис. 1, 2).

В таблице 2 представлены оценки параметров для радиобиологических данных из статьи [1], а также показано согласие с эмпирическим распределением. При уровне значимости $\alpha = 0.05$ получаем согласие для $16/19 \cdot 100\% = 84\%$ случаев.

Наблюдаемое количество аномалий определяется в целом двумя факторами: их исходной распространенностью и интенсивностью их образования в процессе митоза. За увеличение исходной распространенности отвечает параметр q логарифмического распределения, а за интенсивность образования аномалий в процессе митоза параметр λ пуассоновского распределения. Поскольку распределения суммы и самих случайных величин однопараметри-

In vivo					In vitro						
Гр	λ	q	p-v	Гр	λ	q	p-v	Гр	λ	q	p-v
0	0.38	5.9e-7	0.13	0	0.39	3.7e-6	0.59	5	0.67	3.3e-7	0.81
5	0.67	3.3e-7	0.81	5	0.14	0.80	0.73	10	0.83	6.7e-7	0.21
10	0.83	6.7e-7	0.21	10	0.59	1.1e-7	0.03	15	1.15	5.3e-7	0.01
15	1.15	5.3e-7	0.01	15	0.41	0.76	0.36	20	1.71	1e-5	0.03
20	1.71	1e-5	0.03	20	0.37	0.56	0.45	25	1.04	0.07	0.55
25	1.04	0.07	0.55	25	0.88	0.37	0.22	30	1.48	0.33	0.33
30	1.48	0.33	0.33	30	0.78	0.53	0.26	35	1.75	0.19	0.11
35	1.75	0.19	0.11	35	1.10	0.43	0.43	40	2.05	0.22	0.16
40	2.05	0.22	0.16	40	1.46	0.29	0.38	45	2.36	0.18	0.08

Таблица 2: Оценки параметров и значимости критерия хи-квадрат по данным *in vivo* и *in vitro* для логарифмически-пуассоновского распределения.

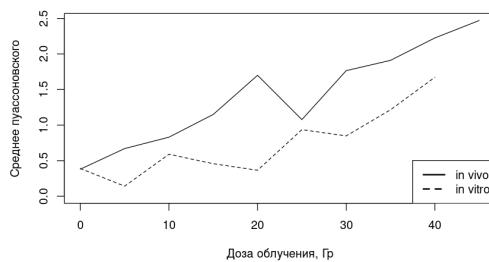


Рис. 3: Оценка параметра λ в зависимости от дозы облучения

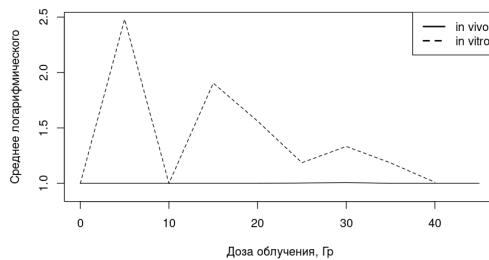


Рис. 4: Среднее лог. распределения в зависимости от дозы облучения

ческие, при интерпретации параметров можно опираться на их средние значения.

Динамика оценок параметра λ , соответственно средних значений, свидетельствует о положительной линейной зависимости их от дозы облучения, что очевидно, и о значимо меньших значениях в эксперименте *in vitro* (рис. 3), так как выжившие клетки обладают большим иммунитетом.

Что касается исходной распространенности аномалий, то, согласно динамике оценок параметра q в зависимости от дозы облучения (рис. 4), зависимости от дозы облучения нет, а в эксперименте *in vivo* базовая распространенность практически не выражена и существенно меньше, чем в эксперименте *in vitro*. Это объясняется тем, что от дозы облучения зависит только количество выживших клеток, а не распространность их аномалий, которая, судя по графику, очень вариабельна, но существенно выше базовой распространенности до начала эксперимента.

Обобщение на случай произвольных распределений

Рассмотрим модель сложного распределения $\zeta_\tau = \xi_1 + \dots + \xi_\tau$, в котором случайное число слагаемых τ распределено по (P_0, P_1, P_2, \dots) , а независимые компоненты по (p_0, p_1, p_2, \dots) . Вероятности сложного закона распределения можно выразить через конечную суммы

$$P\{\zeta_\tau = n\} = \frac{1}{n!} \sum_{k=1}^n \Theta_k G_n^k, \quad \text{где } \Theta_k = \sum_{i=k}^{\infty} P_i C_i^{i-k} p_0^{i-k},$$

$$G_n^k = n! \sum_{\sum_i^k n_i = n} \prod_{i=1}^k p_{n_i}$$

и $n = (n_1, \dots, n_k)$, $n_i > 0$, упорядоченные разбиения натурального числа n . Сравнивая выражения G_n^k в случае логарифмических и пуассоновских независимых величин, приходим к двум видам суммирования над упорядоченными разбиениями:

$$u_n^k = \sum_{\sum_i^k n_i = n, n_i > 0} \frac{n!}{n_1 \dots n_k}, \quad v_n^k = \sum_{\sum_i^k n_i = n, n_i > 0} \frac{n!}{n_1! \dots n_k!}$$

с рекуррентными соотношениями

$$u_n^k = k u_{n-1}^{k-1} + (n-1) u_{n-1}^k, \quad v_n^k = k (v_{n-1}^{k-1} + v_{n-1}^k),$$

которые приводят к существенным частным случаям в виде чисел Стирлинга первого и второго рода $s(n, k) = \frac{1}{k!} u_n^k$ и $S(n, k) = \frac{1}{k!} v_n^k$. В случае пуассоновского распределения независимых компонент сумма по упорядоченным разбиениям имеет тип v_n^k , а в случае логарифмического тип u_n^k .

При логарифмическом распределении (2) числа компонент τ имеем $\Theta_k = \frac{\alpha}{k} \left(\frac{q}{1-q\rho_0} \right)^k$, а при пуассоновском $\Theta_k = \frac{\lambda^k}{k!} e^{-\lambda+\lambda\rho_0}$. Отличие в константах $\frac{1}{k}$ или $\frac{1}{k!}$ приводит к разному виду коэффициентов. Например, пуассоновское τ с внутренним логарифмическим дает сочетание $\frac{1}{k!} u_n^k$, приводящее напрямую к степенному ряду с коэффициентами в виде чисел Стирлинга первого рода, соответственно к убывающему факториалу и к вероятностям отрицательного биномиального распределения. Реинтрантный пуассон приводит к вероятностям с весами в виде чисел Стирлинга второго рода, реинтрантный логарифм или логарифм-пуассон дадут соответственно взвешенные числа Стирлинга первого и второго рода вида $(k-1)!s(n, k)$ и $(k-1)!S(n, k)$.

Заключение

Конечные суммы над упорядоченными разбиениями, связанные с числами Стирлинга, позволяют систематизировать разные виды сложных распределений и тем самым ускорить процесс выбора наиболее адекватной модели.

Список литературы

- [1] Динамика роста числа ядерных аномалий рабдомиосаркомы RA-23 при увеличении дозы острого редкоионизирующего облучения. Исследование на основе модели реинтрантно-биномиального распределения / Алексеева Н. П., Алексеев А. О., Вахтин Ю. Б., Кравцов В. Ю., Кузоватов С. Н. и Скорикова Т. И. // Цитология. 2008. — С. 528–534.
- [2] Феллер В. Введение в теорию вероятностей и её приложения. В 2 т. Москва : Мир, 1952. Т. 1.
- [3] Грэхем Р., Кнут Д., Паташник О. Конкретная математика. Основание информатики. Москва : Мир, 1998. ISBN: 5-03-001793-3.

Вычислительные задачи механики и астрономии



Кустова Елена Владимировна

д.ф.-м.н., профессор, и.о. декана математико-механического факультета,
заведующая кафедрой гидроаэромеханики



Савченко Сергей Сергеевич

к.ф.-м.н., старший научный сотрудник кафедры астрофизики

Сферическая адаптация MUSCL схемы для решения задачи растекающихся слоёв нейтронных звёзд

Русаков А.С., СПбГУ, Санкт-Петербург st075810@student.spbu.ru,
 Аболмасов П.К., Tel Aviv University, Raymond & Beverly Sackler School of Physics and
 Astronomy pavel.abolmasov@gmail.com

Аннотация

При рассмотрении аккрецирующих двойных систем с нейтронной звездой возникает задача о тангенциальном торможении плазмы из аккреционного диска о поверхность нейтронной звезды. Вещество вращающееся с кеплеровской скоростью тормозит до скорости вращения поверхности нейтронной звезды, при этом образуя тонкий аккреционный пограничный слой. Численное решение подобной задачи требует разработки алгоритма решения уравнений газодинамики на сферической сетке. В данной работе представлена консервативная схема второго порядка для уравнений газодинамики на произвольной сферической сетке.

Введение

Большинство наблюдаемых нейтронных звёзд – объекты в двойных системах, аккрецирующие посредством аккреционного диска. Значительная часть их светимости исходит из так называемого пограничного слоя, где плазма из аккреционного диска тормозит до скорости вращения поверхности нейтронной звезды. Физика данного процесса была описана в статье [2]. Плотность потока рентген излучения нейтронных звёзд настолько велика, что может уравновесить разность силы гравитации и центробежной силы. Ввиду того, что пограничный слой тонкий, задачу о его распространении по поверхности нейтронной звезды можно решать на сферической сетке, что делает её двухмерной. Также это позволит рассматривать случаи геометрии, в которых ось вращения аккреционного диска и нейтронной звезды не совпадают, как, например на Рис. 1. При этом поток вещества сверхзвуковой, и может достигать скоростей в $10M$ и выше. В одном из предложенных численных решений в [1] использовался спектральный метод на основе сферических гармоник, однако он посредственно работает со сверхзвуковым потоком. Другой подход, предполагает использование координатных сеток и разностных схем, однако стандартные координатные сетки будут иметь сингулярности у полюсов.

Для решения этих проблем было решено адаптировать существующую двухмерную MUSCL схему [4] для сжимаемой газодинамики с использованием приближённых методов решения задачи Римана на симметричную сферическую сетку.

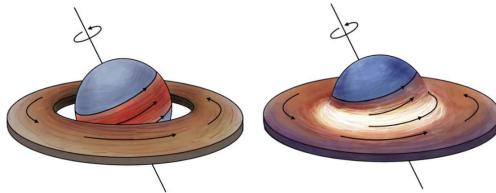


Рис. 1: Варианты геометрии задачи из [5]. Левое изображение содержит только растекающийся слой (красная часть у поверхности), правое дополнительно содержит пограничный слой между диском и поверхностью.

Адаптация численной схемы

В качестве основы был взят метод MUSCL(Monotonic Upstream-centered Scheme for Conservation Laws) реконструкции на произвольных сетках из [4]. Основой метода является реконструкция высокого порядка на грани ячеек. Поскольку алгоритм представлен для произвольных сеток, он был реализован в общем виде, что позволило протестировать разные сферические дискретизации, основанные на правильных многогранниках как на Рис. 2. Использование подобных сеток позволило избежать проблем у полюсов ввиду симметрии. Исключением стала UV Sphere сетка, она не использовалась, поскольку имеет выделенные полюса и не симметрична относительно поворотов.

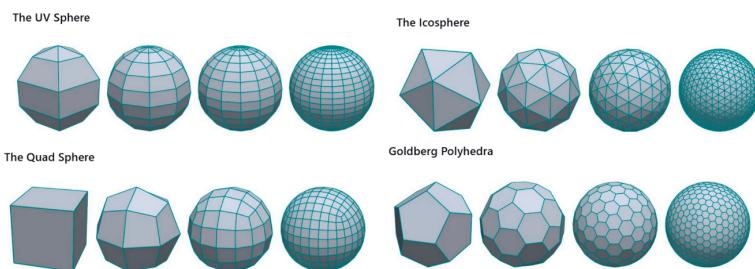


Рис. 2: Виды сферических сеток из [3].

Двухмерная MUSCL схема для произвольных сеток

Рассмотрим скалярное гиперболическое уравнение с начальной задачей:

$$\partial_t u(\mathbf{x}, t) + \nabla \mathbf{F}(\mathbf{x}, t) = 0 \quad u(\mathbf{x}, t = 0) = u_0. \quad (1)$$

Пусть Ω – область определения, $x \in \Omega$. Рассмотрим дискретизацию Ω на многоугольники K_i с произвольным числом граней. Обозначим за $\mathcal{V}(i)$ множество соседних граней к K_i , которые имеют общее ребро $S_{ij} = K_i \cap K_j$. U_i^n – дискретное значение переменной u в центре грани K_i – точке \mathbf{B}_i на момент $t_n = t_0 + n\Delta t$. Тогда численная схема может быть записана в виде:

$$U_i^{n+1} = U_i^n - \Delta t \sum_{\mathcal{V}(i)} \frac{S_{ij}}{K_i} \phi(U_{ij}^n, U_{ji}^n, \mathbf{n}_{ij}). \quad (2)$$

Тут S_{ij} и K_i – длина ребра и площадь грани соответственно, U_{ij}^n, U_{ji}^n – реконструированные на грани S_{ij} значения. \mathbf{n}_{ij} – нормаль к ребру, ϕ – функция приближённого решения задачи Римана (HLLE, HLLC и др.). Для реконструкции значений в центре ребра \mathbf{M}_{ij} , необходимо найти передний и задний наклоны p_{ij}^+ и p_{ij}^- сохраняющейся величины. Тогда на момент t_n (опуская индекс n):

$$\begin{aligned} U_{ij} &= U_i + p_{ij}^+ \varphi(r_{ij}, G_{ij}) \|\mathbf{B}_i \mathbf{M}_{ij}\| \\ U_{ji} &= U_j + p_{ji}^+ \varphi(r_{ji}, G_{ji}) \|\mathbf{B}_j \mathbf{M}_{ij}\|. \end{aligned} \quad (3)$$

Где $r_{ij} = p_{ij}^- / p_{ij}^+$, φ – функция-ограничитель, превращающая два угла наклона величин в один ограниченный для того чтобы не допускать нефизических осцилляций, G_{ij} – набор геометрических параметров сетки.

При особом выборе передней и задней точек \mathbf{H}_{ij}^+ и \mathbf{H}_{ij}^- наклоны можно выразить как:

$$p_{ij}^+ = \frac{U_{H_{ij}^+} - U_i}{\|\mathbf{B}_i \mathbf{H}_{ij}^+\|} \quad p_{ij}^- = \frac{U_i - U_{H_{ij}^-}}{\|\mathbf{B}_i \mathbf{H}_{ij}^-\|}. \quad (4)$$

Процесс нахождения точек \mathbf{H}_{ij}^+ и \mathbf{H}_{ij}^- сильно зависит от сетки. На Рис. 3 представлен пример решения задачи на плоскости. Рассмотрим центральную грань с центром в \mathbf{B}_i . Проведём прямую, соединяющую эту точку с центром рассматриваемой грани \mathbf{M}_{ij} (зелёный пунктир). Среди всех соседних ячеек, включая тех, которые граничат только одной точкой, выберем ту, угол между направлением на центр которой и ранее определённой прямой будет минимальен. Обозначим эту точку $\mathbf{B}_{ij_1}^+$. Аналогично найдём $\mathbf{B}_{ij_2}^+$ с дополнительным условием, что она должна быть с другой стороны от прямой $\mathbf{B}_i \mathbf{M}_{ij}$.

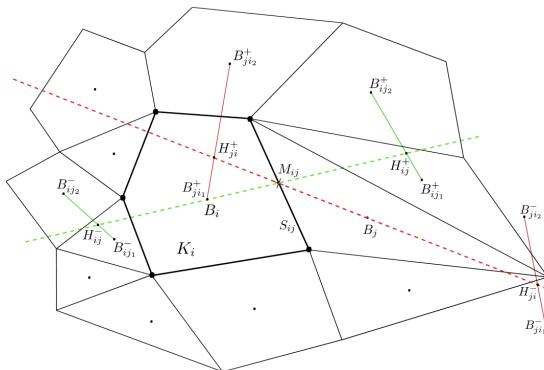


Рис. 3: Вычисление H_{ij}^+ и H_{ij}^- из [4].

Точку пересечения прямых $B_i M_{ij}$ и $B_{ij1}^+ B_{ij2}^+$ назовём H_{ij}^+ . Значение сохраняющейся величины в ней получим, интерполировав значения между ранее выбранными гранями с центрами в B_{ij1}^+ и B_{ij2}^+ . Аналогично найдём задние точки и значения переменных в H_{ij}^- , а затем повторим алгоритм для ячейки с центром в B_j (прямая с красным пунктиром). После всей этой процедуры получим углы наклона переменных, а из них значения интерполированных величин U_{ij} и U_{ji} по формуле 3.

Реализация данной схемы на сетке потребовала изменений по сравнению с аналогичной схемой на плоскости. Были испытаны 2 подхода: использование сетки как многогранника и как сферической тесселяции. При первом подходе некоторые прямые на Рис. 3 становятся ломанными, расстояния нужно искать как сумму длин элементов ломаной.

При сферической тесселяции многоугольники из плоских становятся сферическими, а прямые становятся дугами больших кругов. Были испытаны оба подхода, лучше всего себя проявил сферический, к тому же он является более правильным с физической точки зрения.

Газодинамика и тесты

Все тесты производились на двух системах уравнений: адиабатической и изотермической. Сохраняющиеся величины: Σ – вертикально интегрированная плотность, \mathbf{l} – момент импульса, Σe – полная энергия. Примитивная переменная Π – вертикально интегрированное давление. Плотность и давление интегрируются вдоль радиуса, а угловой момент содержит 3 компоненты.

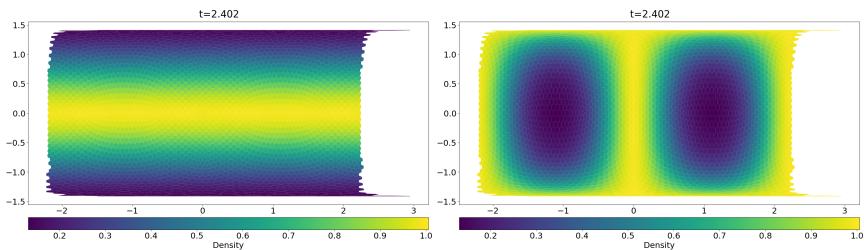


Рис. 4: Проекции плотности при твердотельном вращении.

В приближении тонкого слоя мы считаем, что радиальная компонента скорости равна 0, что даёт нам дополнительное уравнение связи $(\mathbf{r}, \mathbf{v}) = 0$. Это позволяет упростить восстановление скорости как примитивной переменной до $\mathbf{v} = \frac{\mathbf{r} \times \mathbf{l}}{m|\mathbf{r}|^2}$.

Адиабатическая система уравнений представлена уравнением 1, где:

$$u = \begin{pmatrix} \Sigma \\ 1 \\ \Sigma \cdot e \end{pmatrix} \quad \mathbf{F} = \begin{pmatrix} \Sigma(\mathbf{n} \cdot \mathbf{v}) \\ 1(\mathbf{n} \cdot \mathbf{v}) - (\mathbf{n} \times \mathbf{R})\Pi \\ (\Sigma \cdot e + \Pi)(\mathbf{n} \cdot \mathbf{v}). \end{pmatrix} \quad (5)$$

Уравнение состояния:

$$e = \frac{\mathbf{v}^2}{2} + \frac{\Pi}{(\gamma - 1)\Sigma}. \quad (6)$$

Где γ – показатель адиабаты. Изотермическая система уравнений получается из адиабатической исключением третьего уравнения, а также заменой уравнения состояния на $\Pi = a^2\Sigma$.

Тесты

Одной проблемой, которой данная схема должна избегать это изменение поведения ввиду особенностей сетки. Простой тест на твердотельное вращение с разными осями может показать, что наш алгоритм достаточно симметричен и не имеет особенностей в полюсах. В изотермической системе уравнений было рассмотрено вращение вокруг двух разных осей с одинаковыми начальными распределениями плотности и одинаковой угловой скоростью. Результаты представлены на Рис. 4. Оба решения ведут себя одинаково и имеют одинаковые профили плотности на одинаковые моменты времени.

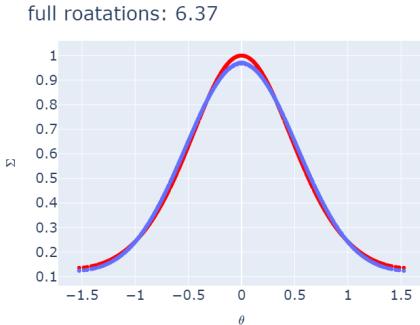


Рис. 5: Красные точки – аналитика, синие – численный профиль.

Другим важным тестом оказалось сохранение аналитических профилей сохраняющихся величин. При твердотельном вращении и изотермической системе уравнений должен сохраняться профиль плотности:

$$\rho = \rho_0 e^{\frac{M^2}{2} \sin^2 \theta} \quad M = \Omega R/a. \quad (7)$$

Тут Ω – угловая скорость твердотельного вращения, a – изотермическая скорость звука θ – полярный угол.

На Рис. 5 представлен численный и аналитический профили через 6 оберотов при сверхзвуковом твердотельном вращении. Расхождения с аналитической небольшие и вызваны ошибками, связанными с размером ячейки сетки.

Ещё одним интересным тестом стала неустойчивость Кельвина-Гельмгольца при адиабатической системе уравнений. Для её рассмотрения были взяты начальные условия, где сфера разделена вдоль экватора на две половины, вращающиеся в разные стороны в дозвуковом режиме. В результате хорошо наблюдались характерные спирали на Рис. 6, особенно хорошо проявляющие себя на графике радиальной компоненты завихрённости $\omega = \nabla \times \mathbf{v}$.

Заключение

В данной работе был представлен алгоритм решения газодинамических уравнений на сфере в применении к астрофизической задаче. На данный момент продолжаются тесты реализации алгоритма для выявления потенциальных ошибок. Для получения астрофизических результатов потребуется реализация функции источников, а также различные поправки, связанные с физикой нейтронных звёзд.

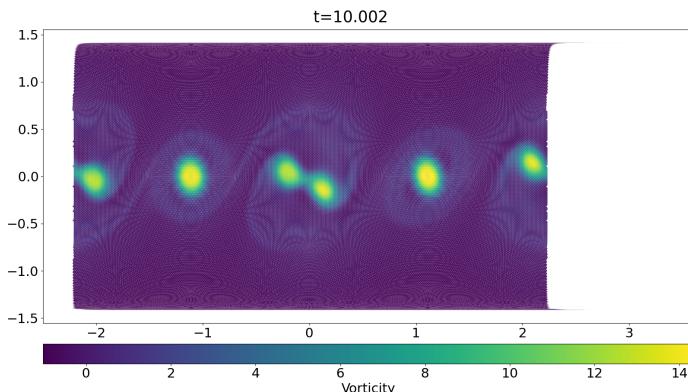


Рис. 6: Радиальная компонента завихрённости скорости. Муаровый узор вызван сжатием изображения очень мелкой сетки.

Список литературы

1. *Abolmasov P., Näyttälä J., Poutanen J.* Kilohertz quasi-periodic oscillations from neutron star spreading layers // Astronomy and Astrophysics. — 2020. — Июнь. — Т. 638. — А142—А142.
2. *Inogamov N. A., Sunyaev R. A.* Spread of matter over a neutron-star surface during disk accretion // AstL. — 1999. — Апр. — Т. 25. — С. 269—293.
3. *Sieger D.* Generating Meshes of a Sphere. — Daniel Sieger, 03.2021.
4. *Touze C. L., Murrone A., Guillard H.* Multislope MUSCL method for general unstructured meshes // Journal of Computational Physics. — 2015. — Март. — Т. 284. — С. 389—418.
5. X-Ray Polarized View on the Accretion Geometry in the X-Ray Binary Circinus X-1 / J. Rankin [и др.] // arXiv (Cornell University). — 2023. — Нояб.

Создание программного продукта для моделирования химической релаксации газа за ударной волной

Зернов С. Н., СПбГУ, Санкт-Петербург st095228@student.spbu.ru

Аннотация

В данной работе описывается численное моделирование макропараметров газа за ударной волной с учетом неравновесных химических реакций диссоциации. Для расчета макропараметров смеси O_2/O была использована система уравнений Чепмена-Энскога [1] в нулевом приближении в сочетании с однотемпературной моделью.

Был разработан прототип на языке Python с возможностью выбора временного интервала, абсолютной точности разбиения и опцией включения профилирования. Последнее позволило обнаружить проблемы с производительностью, которые будут учтены при создании основного программного продукта. Для верификации модели были оценены ошибки в законах сохранения массы, импульса и энергии на каждом шаге решения системы дифференциальных уравнений. Собранные данные показывают, что абсолютная величина ошибки пропорциональна абсолютной точности разбиения и приемлемо мала.

Кроме того, была модифицирована библиотека с открытым исходным кодом КАРРА [2]: обновлена система сборки, версия стандарта C++ и добавлен слой C API. С API позволяет интегрировать стороннее программное обеспечение в новый проект, в нашем случае — специфическую функцию для расчета коэффициентов скорости диссоциации. Эта техника показывает, что возможно повторное использование существующих проектов через межъязыковое взаимодействие с приемлемым влиянием на производительность.

Данная работа дополняет текущие представления о проектировании программных продуктов в сложной междисциплинарной области кинетической теории и высокопроизводительных вычислений, создавая фундамент для будущих проектов.

Список литературы

- [1] Нагнибеда Е. А., Кустова Е. В. Кинетическая теория процессов переноса и релаксации в потоках неравновесных реагирующих газов. – Изд.-во Санкт-Петербург. ун-та, 2003.
- [2] L. Campoli, G.P. Oblapenko, E.V. Kustova, KAPPA: Kinetic approach to physical processes in atmospheres library in C++, Computer Physics Communications, Volume 236, 2019, Pages 244-267, <https://doi.org/10.1016/j.cpc.2018.10.016>.

Модели, методы и приложения тропической математики



Кривулин Николай Кимович

д.ф.-м.н., профессор, профессор кафедры статистического
моделирования

Оценка альтернатив на основе парных сравнений в задаче об определении лидера группы

Филатова А.А., студентка 4-го курса бакалавриата СПбГУ
rii.filatova@gmail.com,

Кривулин Н.К., д.ф.-м.н. профессор кафедры статистического моделирования
 СПбГУ *nkk@math.spbu.ru*

Аннотация

Рассматривается многокритериальная задача оценки альтернатив на основе парных сравнений, которая возникает при определении лидера группы. Для ее решения применяются методы анализа иерархий, взвешенных геометрических средних и log-чебышевской аппроксимации. Полученные результаты сравниваются между собой.

Введение

Один из ключевых компонентов, который играет определяющую роль в достижении целей программного проекта и является залогом успешной командной работы – это правильный выбор лидера. При выборе руководителя следует учитывать несколько факторов, а значит принятие данного решения можно рассматривать как многокритериальную задачу оценки альтернатив на основе парных сравнений по некоторым неравноценным критериям.

В данной задаче n альтернатив сравниваются попарно по m критериям. Полученные результаты парных сравнений представляют в виде матрицы A_k размерности n , в соответствии с критерием $k = 1, \dots, m$. Критерий также сравниваются попарно, и результаты этих сравнений образуют матрицу C . Необходимо на основе матриц парных сравнений A_1, \dots, A_m и C определить степень предпочтения каждой альтернативы. Основная трудность при решении заключается в отсутствии единого решения, оптимального по всем критериям одновременно.

Есть целый ряд методов решения данных задач, как эвристических, (например, метод анализа иерархий Т. Саати [1]), так и аналитических (метод взвешенных геометрических средних [2], метод log-чебышевской аппроксимации [3]). Учитывая, что результаты, полученные одним методом, могут значительно отличаться от результатов другого метода [4], возникает необходимость в сравнении этих результатов для получения более разумного решения. Если результаты, полученные разными методами, практически совпадают, это свидетельствует о близости решений к оптимальному.

Постановка задачи определения лидера группы

Рассматривается задача выбора руководителя инженерной команды [5], в которой исследуются 4 кандидата (альтернативы) на должность лидера группы ((1), (2), (3), (4)). Претенденты сравниваются попарно по 4 критериям: личные качества, академические достижения, опыт работы в команде, уровень компетентности разработчика (грейд).

Результаты парных сравнений альтернатив по каждому критерию описываются матрицами:

$$A_1 = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 1/2 & 1 & 3 & 2 \\ 1/3 & 1/3 & 1 & 1/3 \\ 1/4 & 1/2 & 3 & 1 \end{pmatrix}, \quad A_2 = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 1/2 & 1 & 2 & 3 \\ 1/3 & 1/2 & 1 & 2 \\ 1/4 & 1/3 & 1/2 & 1 \end{pmatrix},$$

$$A_3 = \begin{pmatrix} 1 & 2 & 2 & 3 \\ 1/2 & 1 & 2 & 3 \\ 1/2 & 1/2 & 1 & 2 \\ 1/3 & 1/3 & 1/2 & 1 \end{pmatrix}, \quad A_4 = \begin{pmatrix} 1 & 3 & 2 & 3 \\ 1/3 & 1 & 2 & 4 \\ 1/2 & 1/2 & 1 & 1 \\ 1/3 & 1/4 & 1 & 1 \end{pmatrix},$$

а результаты парных сравнений критериев — матрицей:

$$C = \begin{pmatrix} 1 & 4 & 3 & 7 \\ 1/4 & 1 & 1/3 & 3 \\ 1/3 & 3 & 1 & 5 \\ 1/7 & 1/3 & 1/5 & 1 \end{pmatrix}.$$

Необходимо определить вектор рейтингов альтернатив x , который устанавливает порядок на множестве альтернатив. Решение находится тремя методами: методом анализа иерархий, методом взвешенных геометрических средних и методом log-чебышевской аппроксимации.

Метод анализа иерархий

Традиционный способ решения задачи оценки альтернатив на основе их парных сравнений – применение метода анализа иерархий Т. Саати. Он заключается в составлении взвешенной суммы нормированных главных собственных векторов матриц A_1, \dots, A_m , где в качестве весов выступают элементы нормированного главного собственного вектора матрицы C .

Нормированный главный собственный вектор матрицы C равен

$$\mathbf{w} \approx (0,5476 \ 0,1265 \ 0,2700 \ 0,0559)^T.$$

Нормированные главные собственные векторы матриц A_1, A_2, A_3, A_4 имеют следующий вид:

$$\begin{aligned}\mathbf{x}_1 &\approx (0,4688 \ 0,2684 \ 0,0947 \ 0,1681)^T, \\ \mathbf{x}_2 &\approx (0,4673 \ 0,2772 \ 0,1601 \ 0,0954)^T, \\ \mathbf{x}_3 &\approx (0,4155 \ 0,2926 \ 0,1850 \ 0,1069)^T, \\ \mathbf{x}_4 &\approx (0,4568 \ 0,2799 \ 0,1489 \ 0,1144)^T.\end{aligned}$$

Тогда вектор рейтингов:

$$\mathbf{x} = w_1 \mathbf{x}_1 + w_2 \mathbf{x}_2 + w_3 \mathbf{x}_3 + w_4 \mathbf{x}_4 \approx (0,45355 \ 0,27669 \ 0,13038 \ 0,13938)^T.$$

Вектор рейтингов, нормированный относительно максимального элемента, равен

$$\mathbf{x}_{\text{АНР}} \approx (1 \ 0,6101 \ 0,2875 \ 0,3073)^T.$$

Метод взвешенных геометрических средних

Прямой способ решения многокритериальной задачи оценки альтернатив заключается в аппроксимации матриц парных сравнений по всем критериям общей согласованной матрицей. Применение логарифмической шкалы для аппроксимации позволяет получить аналитическое решение задачи непосредственно в явной форме. В случае решения задачи с помощью логевклидовой аппроксимации полученный вектор \mathbf{x} называют решением многокритериальной задачи методом взвешенных геометрических средних.

Для нахождения решения нужно вычислить нормированный относительно суммы элементов вектор геометрических средних $\mathbf{w} = (w_i)$ матрицы C парных сравнений критериев. Далее необходимо определить векторы \mathbf{x}_i геометрических средних матриц A_i парных сравнений альтернатив. Заметим, что геометрическое среднее вычисляется по строкам матрицы.

Элементы каждого вектора $\mathbf{x}_i = (x_{j,i})$ возводятся в степень w_i , а затем результат перемножения элементов этих векторов, соответствующих индексу j , берется в качестве элемента вектора рейтингов \mathbf{x} .

Нормированный вектор геометрических средних матрицы C выражается следующим образом:

$$\mathbf{w} \approx (0,5462 \ 0,1276 \ 0,2698 \ 0,0564)^T.$$

Векторы геометрических средних для матриц A_1, A_2, A_3, A_4 парных сравнений:

$$\begin{aligned}\mathbf{x}_1 &\approx (2,2134 \ 1,3161 \ 0,4387 \ 0,7825)^T, \\ \mathbf{x}_2 &\approx (2,2134 \ 1,3161 \ 0,7598 \ 0,4518)^T, \\ \mathbf{x}_3 &\approx (1,8612 \ 1,3161 \ 0,8409 \ 0,4855)^T, \\ \mathbf{x}_4 &\approx (2,0598 \ 1,2779 \ 0,7071 \ 0,5373)^T.\end{aligned}$$

Получим вектор рейтингов:

$$\mathbf{x} = \begin{pmatrix} x_{1,1}^{w_1} \cdot x_{1,2}^{w_2} \cdot x_{1,3}^{w_3} \cdot x_{1,4}^{w_4} \\ x_{2,1}^{w_1} \cdot x_{2,2}^{w_2} \cdot x_{2,3}^{w_3} \cdot x_{2,4}^{w_4} \\ x_{3,1}^{w_1} \cdot x_{3,2}^{w_2} \cdot x_{3,3}^{w_3} \cdot x_{3,4}^{w_4} \\ x_{4,1}^{w_1} \cdot x_{4,2}^{w_2} \cdot x_{4,3}^{w_3} \cdot x_{4,4}^{w_4} \end{pmatrix} \approx (2,1037 \ 1,3139 \ 0,5762 \ 0,6279)^T.$$

После нормирования относительно максимального элемента его можно записать как

$$\mathbf{x}_{\text{WGM}} \approx (1 \ 0,6245 \ 0,2739 \ 0,2985)^T.$$

Метод log-чебышевской аппроксимации

Еще один метод решения, основанный на аппроксимации матриц парных сравнений, – это метод log-чебышевской аппроксимации с использованием тропической оптимизации.

Элементы тропической математики

Тропическая (идемпотентная) математика изучает теорию и приложения алгебраических систем с идемпотентными операциями [6]. При применении такой операции к одному и тому же аргументу результатом является тот же самый аргумент.

Решение задачи парных сравнений будем рассматривать в терминах max–алгебры $\mathbb{R}_{\max} = (\mathbb{R}_+, \max, \times, 0, 1)$, где $\mathbb{R}_+ = \{x \in \mathbb{R} \mid x \geq 0\}$, умножение определено как обычно, а сложение является идемпотентной операцией и определено как максимум (обозначается знаком \oplus).

В векторных и матричных операциях используются обычные правила, но с заменой арифметического сложения на операцию максимума.

След матрицы $\mathbf{A} = (a_{ij})$ размерности $n \times n$ вычисляется по формуле

$$\operatorname{tr} \mathbf{A} = a_{11} \oplus \cdots \oplus a_{nn}.$$

Спектральный радиус \mathbf{A} — скаляр, определяемый формулой

$$\lambda = \operatorname{tr} \mathbf{A} \oplus \cdots \oplus \operatorname{tr}^{1/n} (\mathbf{A}^n).$$

Если $\lambda \leq 1$, то для \mathbf{A} можно построить матрицу Клини

$$\mathbf{A}^* = \mathbf{I} \oplus \mathbf{A} \oplus \cdots \oplus \mathbf{A}^{n-1}.$$

Решение задачи

Найдем решение с помощью процедуры, предложенной в [7].

Определим спектральный радиус матрицы \mathbf{C} , который равняется

$$\lambda = \operatorname{tr} \mathbf{C} \oplus \operatorname{tr}^{1/2} (\mathbf{C}^2) \oplus \operatorname{tr}^{1/3} (\mathbf{C}^3) \oplus \operatorname{tr}^{1/4} (\mathbf{C}^4) = 27^{1/4} / 7^{1/4}.$$

Найдем матрицу Клини, определяющую рейтинг весов критериев:

$$(\lambda^{-1} \mathbf{C})^* = \mathbf{I} \oplus \lambda^{-1} \mathbf{C} \oplus \lambda^{-2} \mathbf{C}^2 \oplus \lambda^{-3} \mathbf{C}^3 = \\ \begin{pmatrix} 1 & 3^{1/2} \cdot 7^{1/2} & 3^{1/4} \cdot 7^{1/4} & 3^{3/4} \cdot 7^{3/4} \\ 3^{-1/2} \cdot 7^{-1/2} & 1 & 3^{-1/4} \cdot 7^{-1/4} & 3^{1/4} \cdot 7^{1/4} \\ 3^{-1/4} \cdot 7^{-1/4} & 3^{1/4} \cdot 7^{1/4} & 1 & 3^{1/2} \cdot 7^{1/2} \\ 3^{-3/4} \cdot 7^{-3/4} & 3^{-1/4} \cdot 7^{-1/4} & 3^{-1/2} \cdot 7^{-1/2} & 1 \end{pmatrix}.$$

Столбцы матрицы Клини коллинеарны — в качестве вектора весов можно взять любой, например,

$$\mathbf{w} = (1 \ 3^{-1/2} \cdot 7^{-1/2} \ 3^{-1/4} \cdot 7^{-1/4} \ 3^{-3/4} \cdot 7^{-3/4})^T.$$

Для нахождения рейтингов альтернатив составим взвешенную тропическую сумму матриц $\mathbf{A}_1, \mathbf{A}_2, \mathbf{A}_3$ и \mathbf{A}_4 в виде

$$\mathbf{B} = w_1 \mathbf{A}_1 \oplus w_2 \mathbf{A}_2 \oplus w_3 \mathbf{A}_3 \oplus w_4 \mathbf{A}_4 = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 1/2 & 1 & 3 & 2 \\ 1/3 & 1/3 & 1 & 2 \cdot 3^{-1/4} \cdot 7^{-1/4} \\ 1/4 & 1/2 & 3 & 1 \end{pmatrix}.$$

Спектральный радиус матрицы \mathbf{B} равен

$$\mu = \text{tr } \mathbf{B} \oplus \text{tr}^{1/2} (\mathbf{B}^2) \oplus \text{tr}^{1/3} (\mathbf{B}^3) \oplus \text{tr}^{1/4} (\mathbf{B}^4) = 2^{1/2} \cdot 3^{3/8} \cdot 7^{-1/8}.$$

Найдем матрицу Клини:

$$\mathbf{B}^* = (\mu^{-1} \mathbf{B})^* = \mathbf{I} \oplus \mu^{-1} \mathbf{B} \oplus \mu^{-2} \mathbf{B}^2 \oplus \mu^{-3} \mathbf{B}^3 =$$

$$\begin{pmatrix} 1 & 2^{1/2} \cdot 7^{1/8} \cdot 3^{-3/8} & 2 \cdot 3^{1/4} \cdot 7^{1/4} & 8^{1/2} \cdot 7^{1/8} \cdot 3^{-3/8} \\ 7^{3/8} \cdot 2^{-1/2} \cdot 3^{-9/8} & 1 & 3^{1/4} \cdot 7^{1/4} & 2^{1/2} \cdot 7^{1/8} \cdot 3^{-3/8} \\ 7^{1/8} \cdot 2^{-1/2} \cdot 3^{-11/8} & 7^{1/4} \cdot 3^{-7/4} & 1 & 2^{1/2} \cdot 3^{-5/8} \cdot 7^{-1/8} \\ 7^{1/4} \cdot 2^{-1} \cdot 3^{-3/4} & 7^{3/8} \cdot 2^{-1/2} \cdot 3^{-9/8} & 3^{5/8} \cdot 7^{1/8} \cdot 2^{-1/2} & 1 \end{pmatrix}.$$

Не все столбцы \mathbf{B}^* коллинеарны, следовательно, необходимо вычислить наилучший и наихудший дифференцирующие векторы. Наилучшим считается столбец данной матрицы, для которого отношение максимального и минимального элементов является наибольшим. Соответственно наихудший – тот столбец, где это отношение наименьшее.

Вычислим их с помощью формул, приведенных в статье [3].

Для определения наилучшего дифференцирующего вектора найдем среди нормированных столбцов $\mathbf{B}^* = (\mathbf{b}_j^*)$ следующие:

$$\mathbf{x}_k = \mathbf{b}_k^* \|\mathbf{b}_k^*\|^{-1}, \quad k = \arg \max_{1 \leq j \leq n} \|\mathbf{b}_j^*\| \left\| (\mathbf{b}_j^*)^- \right\|.$$

Последовательно вычислим:

$$\|\mathbf{b}_1^*\| \left\| (\mathbf{b}_1^*)^- \right\| = \|\mathbf{b}_2^*\| \left\| (\mathbf{b}_2^*)^- \right\| = 7^{-1/8} \cdot 2^{1/2} \cdot 3^{11/8},$$

$$\|\mathbf{b}_3^*\| \left\| (\mathbf{b}_3^*)^- \right\| = \|\mathbf{b}_4^*\| \left\| (\mathbf{b}_4^*)^- \right\| = 2 \cdot 3^{1/4} \cdot 7^{1/4}.$$

Максимальное значение соответствует $k = 1$ и $k = 2$, получим, что

$$\mathbf{x}_1 \approx (1 \ 0,4262 \ 0,1991 \ 0,3568)^T, \quad \mathbf{x}_2 \approx (1 \ 0,8371 \ 0,1991 \ 0,3568)^T.$$

Наилучшим дифференцирующим вектором будет вектор с наименьшими координатами. В данном случае им является $\mathbf{x}_1 = \mathbf{x}'_{\text{LCA}}$, так как он лучше различает первую и вторую альтернативы.

Наихудший вектор вычисляется с помощью следующей формулы и определяется однозначно:

$$\mathbf{x}_{\text{LCA}}'' = (\mathbf{1}^T (\mu^{-1} \mathbf{B})^*)^- \approx (1 \ 0,8370 \ 0,2336 \ 0,4185)^T.$$

Заключение

В результате решения задачи об определении лидера группы на основе парных сравнений альтернатив (1), (2), (3), (4) по 4 критериям было получено 4 решения: $\boldsymbol{x}_{\text{AHD}}$ методом анализа иерархий, $\boldsymbol{x}_{\text{WGD}}$ методом взвешенных геометрических средних, наилучший $\boldsymbol{x}'_{\text{LCA}}$ и наихудший $\boldsymbol{x}''_{\text{LCA}}$ дифференцирующие векторы методом log-чебышевской аппроксимации. Решения задают одинаковый порядок альтернатив:

$$(1) \succ (2) \succ (4) \succ (3).$$

Таким образом, найденный вектор можно считать оптимальным.

Список литературы

- [1] Саати Т. Принятие решений. Метод анализа иерархий // Пер. с англ. Вачнадзе Р. Г. М.: Радио и связь, 1993. 315 с.
- [2] Crawford G., Williams C. A note on the analysis of subjective judgment matrices // J. Math. Psych. 1985. Vol. 29, N 4. P. 387–405.
- [3] Krivulin N. Application of tropical optimization for solving multicriteria problems of pairwise comparisons using log-Chebyshev approximation // Int. J. Approx. Reason. 2024. Vol. 169, P. 109168.
- [4] Tran N. M. Pairwise ranking: Choice of method can produce arbitrarily different rank order // Linear Algebra Appl. 2013. Vol. 438, N 3. P. 1012–1024.
- [5] Muhsin Z. A. A., Omar M., Ahmad M., Muhsin S. A. Team leader selection by using an Analytic Hierarchy Process (AHP) technique // J. Softw. 2015. Vol. 10, N 10. P. 1216–1227.
- [6] Кривулин Н. К. Методы идемпотентной алгебры в задачах моделирования и анализа сложных систем // СПб.: Изд-во С.-Петербург. ун-та, 2009. 255 с.
- [7] Krivulin N., Sergeev S. Tropical implementation of the Analytical Hierarchy Process decision method // Fuzzy Sets and Systems. 2019. Vol. 377. P. 31-51.

Параллельные алгоритмы, вэйвлетная обработка числовых потоков



Макаров Антон Александрович

д.ф.-м.н., профессор, заведующий кафедрой параллельных алгоритмов

Квазилинейная интерполяция минимальными сплайнами

Лившиц Л.П., СПбГУ, Санкт-Петербург st088046@student.spbu.ru

Аннотация

Данная работа посвящена исследованию метода сплайн-интерполяции, который может использоваться в задачах анализа данных, в задачах навигации, множестве задач представления данных (подробнее см. [1]). Задачи обработки экспериментальных данных сегодня решаются большей частью с применением различных типов сплайнов [2, 3]. В то же время нельзя не упомянуть, что методы, использующие полиномиальные сплайны, не всегда дают приемлемые результаты [4, 5]. Так для решения задач с функциями высоких градиентов классические методы оказываются бессильны, а погрешность кусочно-линейной интерполяции – слишком большой. Такого вида задачи возникают, например, при решении задач в пограничном слое [6, 7]. Предложенный метод квазилинейной сплайн-интерполяции является обобщением метода кусочно-линейной сплайн-интерполяции, однако получаемая при использовании данного метода ошибка в ряде случаев оказывается меньше, чем при применении известных подходов. В работе получены формулы для координатных сплайнов, доказаны теоремы об ошибке интерполяции функции и её производной. Теоретически полученные результаты подтверждены численными экспериментами. Рассмотрены возможности распараллеливания предложенного метода. Представлена параллельная реализация. Рассчитаны метрики: ускорение и эффективность. Полученные результаты говорят о перспективности масштабирования метода на системы с большим числом потоков и следуют из локальности предложенной схемы.

Список литературы

- [1] Лившиц Л.П., Макаров А.А., Макарова С.В. О квазилинейной интерполяции минимальными сплайнами // Зап. научн. сем. ПОМИ. **524** (2023), 94–111.

- [2] Дем'янович Ю.К., Михлин С.Г. О сеточной аппроксимации функций со-
болевских пространств // Зап. науч. семинаров ЛОМИ АН СССР, **35**
(1973), 6–11.
- [3] Макаров А.А. О построении сплайнов максимальной гладкости // Пробл.
матем. анал. **60** (2011), 25–38.
- [4] Makarov A.A. On example of circular arc approximation by quadratic minimal
splines // Poincare J. Anal. Appl. **2(II)**, 2018, 103–107.
- [5] Kulikov E.K., Makarov A.A. On approximation by hyperbolic splines // J.
Math. Sci. **240:6** (2019), 822–832.
- [6] Куликов Е.К., Макаров А.А. О приближённом решении одной сингулярно
возмущённой краевой задачи // Дифф. ур. проц. 1 (2020), 91–102.
- [7] Kulikov E., Makarov A. On biorthogonal approximation of solutions of some
boundary value problems on Shishkin mesh // AIP Conf. Proc. **2302** (2020),
110005.

О приближении кусочно-постоянными финитными функциями и их обобщениями

Битецк В.О., СПбГУ, Санкт-Петербург st087598@student.spbu.ru

Аннотация

В докладе в рамках теории минимальных сплайнов рассмотрена аппроксимация кусочно-постоянными минимальными сплайнаами нулевого порядка и их неполиномиальными обобщениями, которые были введены в работе [1]. Общий способ построения рассматриваемых минимальных сплайнов заданной гладкости представлен, например, в работе [2]. Интерес к сеточной аппроксимации функций связан с разработкой теории метода конечных элементов (подробнее см. работы [3, 4, 5]).

В данной работе сформулированы и доказаны теоремы об оценках ошибки приближения кусочно-непрерывными минимальными сплайнами. Приведены результаты численных экспериментов, которые согласуются с полученными теоретическими оценками. Разработан программный комплекс для построения аппроксимации кусочно-непрерывными сплайнами и ее визуализации.

Список литературы

- [1] Макаров А.А. Кусочно-непрерывные сплайн-вэйвлеты на неравномерной сетке // Тр. СПИИРАН. **14** (2010), 103–131.
- [2] Макаров А.А. О построении сплайнов максимальной гладкости // Пробл. матем. анал. **60** (2011), 25–38.
- [3] Дем'янович Ю.К., Михлин С.Г. О сеточной аппроксимации функций соллевских пространств // Зап. науч. семинаров ЛОМИ АН СССР, **35** (1973), 6–11.
- [4] Стрэнг Г., Фикс Дж. Теория метода конечных элементов. М., 1977.
- [5] Марчук Г.И., Агошков В.И. Введение в проекционно-сеточные методы. М., 1981.

О курсе «Методика преподавания компьютерных наук» в непедагогических вузах

Евдокимова Т.О., СПбГУ, Санкт-Петербург t.evdokimova@spbu.ru,
Иванцова О.Н., СПбГУ, Санкт-Петербург o.ivancova@spbu.ru

Аннотация

На математико-механическом факультете СПбГУ курсы по методике преподавания читаются на большинстве образовательных программ. Это позволяет студентам овладеть не только своей «рабочей» специальностью, но и, в широком смысле, осознать способы её освоения, в том числе, начальные шаги, которые могут быть осуществлены и в школьном обучении. В связи с этим, закономерен интерес к наличию и реализации аналогичных курсов в других профильных, но не педагогических вузах РФ. В данной работе представлено небольшое исследование, проведенное авторами по этому вопросу.

История вопроса

В СПбГУ с момента открытия образовательной программы «Математическое обеспечение и администрирование информационных систем» для магистров читается обязательный курс «Методика преподавания компьютерных наук».

Ранее, с 2010 года, для студентов 5-го курса специалитета проводился элективный курс «Методика преподавания информатики и ИКТ». Освоение данной дисциплины дает возможность овладеть навыками преподавания, в том числе и для школьников.

Идея разработки и реализации подобного курса была заложена ещё коллегами-математиками, полагающими необходимость профильной подготовки учителей для специализированных школ, которую затруднительно дать в педагогическом вузе.

В связи с этим у авторов возник вопрос о том, реализуются ли подобные курсы в других крупных (ведущих) непедагогических вузах?

Результаты исследования

Были рассмотрены учебные планы следующих вузов:

- Санкт-Петербургский государственный университет

- Московский государственный университет
- Новосибирский государственный университет
- Национальный исследовательский университет ИТМО
- Санкт-Петербургский государственный электротехнический университет «ЛЭТИ»

При этом рассматривались образовательные программы, связанные с вычислительной техникой, программированием и математикой.

Опираясь на данные открытого доступа, удалось выяснить, что в рассматриваемых вузах отсутствуют курсы по методике преподавания компьютерных наук. Анализ проводился по основным образовательным программам как бакалавриата, так и магистратуры.

Другие результаты

Был проведен анализ реализации курсов по методике преподавания компьютерных наук в других непедагогических вузах. В результате получен список следующих вузов:

- Кубанский государственный университет (КубГУ)
- Рязанский государственный университет (РГУ)
- Саратовский государственный университет (СГУ)

Соответствующие дисциплины этих вузов — это «Методика преподавания ИКТ», «Методика преподавания компьютерных наук в высшей школе» и «Методика обучения информатике».

Характеристика курсов

Кубанский государственный университет (КубГУ) реализует дисциплину «Методика преподавания ИКТ» для магистров по направлению подготовки 01.04.02 «Прикладная математика и информатика» по трем образовательным программам:

1. «Математические и информационные технологии в цифровой экономике»
2. «Математическое моделирование в естествознании и технологиях»

3. «Технологии программирования и разработки информационно-коммуникационных систем»

Все образовательные программы предназначены для магистров Факультета компьютерных технологий и прикладной математики трех кафедр:

- Кафедра Прикладной математики
- Кафедра Математического моделирования
- Кафедра Информационных технологий

Рассматриваемая дисциплина является обязательной и читается в первом семестре, продолжительность — 42 ауд. часа. По завершении проводится экзамен. Из аннотации к методическому пособию по данной дисциплине следует, что основное внимание уделяется психолого-педагогическим проблемам преподавателей ИТ; конкретным методикам преподавания программных сред, включая рекомендации по проведению лекционных и семинарских занятий, по конструированию практических заданий и нормативно-правовым вопросам преподавательской деятельности в высшей школе. К сожалению, при более тщательном поиске аннотации и РПД в открытом доступе не найдено!

Рязанский государственный университет (РГУ) реализует дисциплину «Методика преподавания компьютерных наук в высшей школе» для магистров по направлению подготовки 02.04.02 «Фундаментальная информатика и информационные технологии» по образовательной программе «Информационные системы» на Кафедре информатики, вычислительной техники и методики преподавания информатики Института физико-математических и компьютерных наук.

Рассматриваемая дисциплина является элективной и читается в третьем семестре, продолжительность — 54 ауд. часа (лекции – 18, лабораторные работы — 36). По завершении проводится зачет.

Проанализировав РПД можно сказать, что в данной дисциплине основное внимание уделяется методологическим основам методики преподавания компьютерных наук; организационным формам обучения в вузе; контролю и оценке знаний студентов.

В ходе изучения рассматриваемой дисциплины делается акцент на основные задачи, решаемые российской высшей школой при переходе на двухуровневую систему образования, на общее понятие, задачи и функции методики преподавания в высшей школе. Уделяется внимание историческому развитию и становлению современной методической системы, а также взаимосвязи образовательной, воспитательной и развивающей функций обучения. Рассматриваются психологические основы учебного процесса: мотивы

учения студентов, их развитие и формирование; единство преподавания и учения; обучение как сотворчество преподавателя и студентов. Магистрами изучаются современные тенденции развития образования; методологические основы и организация педагогического процесса и инновации в образовании. Подробно рассматриваются такие понятия, как «метод», «прием», «средство» обучения и «педагогическая технология», а также классификация методов обучения. Изучается взаимосвязь методов обучения и условия их оптимального выбора, а также виды педагогических технологий (технологии традиционного обучения, компьютерные технологии, технологии модульного и контекстного обучения, интенсивная технология обучения).

Уделено внимание различным видам контроля в вузе (оперативный, текущий, рубежный, итоговый) и формам проведения: зачеты, экзамены, коллоквиумы, Интернет-экзамены, тестирование, контрольные работы, защиты рефератов, курсовых и дипломных работ, а также особенностям рейтингового контроля и оценки достижений студентов, с учетом его достоинств и недочетов. Рассматриваются и нетрадиционные формы и методы контроля в образовательном процессе.

Лабораторные работы посвящены разработкам модели академического занятия; эффективности методов обучения; использованию средств медиа в обучении; диагностике степени обученности студентов и сформированности профессиональной мотивации студентов.

Саратовский государственный университет (СГУ) реализует дисциплину «Методика преподавания компьютерных наук» для магистров по направлению подготовки 09.04.01 «Информатика и вычислительная техника» по двум образовательным программам: «Сети ЭВМ и телекоммуникации» и «Анализ и синтез распределенных технических систем» на Факультете компьютерных наук и информационных технологий. Рассматриваемая дисциплина является обязательной и читается во втором семестре, продолжительность — 36 ауд. часа (лекции — 28, лабораторные работы — 8 часов). По завершении проводится экзамен.

Проанализировав РПД можно сказать, что в данной дисциплине основное внимание уделяется ИТ-образованию; различным аспектам педагогике и дидактике высшей школы; разнообразным методикам. Подробно рассматриваются такие понятия, как «образование», «образовательное пространство» и «образовательная среда». Изучается специфика ИТ-образования и мировые тенденции его развития. Особое внимание уделяется понятию «педагогика высшей школы». Рассматривается предмет, задачи и методология педагогики высшей школы, а также структура, особенности, закономерности и принципы педагогического процесса в вузе. Изучается специфика педагогического процесса в условиях электронного и дистанционного обучения; включение

бизнес-структур в педагогический процесс подготовки ИТ-специалистов. Говорится об основных документах, регламентирующих педагогический процесс и деятельность преподавателей вузов, о ФГОС ВО, о корпоративных и профессиональных стандартах, о положениях, регламентирующих учебный процесс в вузе. Рассмотрены рабочие учебные планы и рабочие программы дисциплин; тематические планы и учебно-методические комплексы.

Уделено внимание и понятию «дидактика высшей школы». Рассматривается структура обучения студента вуза; теории, концепции и технологии обучения в высшей школе, а также инновационные технологии обучения. Изучается специфика обучения компьютерным наукам студентов разных направлений, возрастов и другие вопросы индивидуализации обучения.

Методическая система обучения студентов компьютерным наукам предполагает изучить методы обучения, виды учебных занятий, средства обучения, специфику средств обучения компьютерным наукам, а также роль лекции, семинара и практических занятий в учебном процессе вуза, методику работы над ними. Даются рекомендации по формированию учебно-методических комплексов (УМК) учебных дисциплин и разработке частных методик для дисциплин компьютерного цикла в соответствии с требованиями ФГОС ВО. Лабораторные работы посвящены созданию собственного электронного курса на основе изученного материала.

В Санкт-Петербургском государственном университете (СПбГУ) курс методики преподавания компьютерных наук является обязательным для магистров второго курса направления «Математическое обеспечение и администрирование информационных систем», состоит из 16 часов лекций и 16 часов практики и завершается зачетом. Курс посвящен знакомству студентов как с методикой преподавания информатики в школе, так и с различными дидактическими подходами, применимыми к большинству дисциплин. Перечислим основные источники, обсуждаемые в этом курсе.

Базовый школьный курс информатики, его цели, задачи, содержание и методика преподавания рассматриваются по учебниками Кушниренко А. Г., Леонова А. Г., Зайдельмана Я. Н., Тарасовой В. В. и по методическим материалам, изложенным в книге Кушниренко А. Г. и Лебедева Г. В. «12 лекций о том, для чего нужен школьный курс информатики и как его преподавать». Дополнительным источником задач и теории к ним является книга Кушниренко А. Г. и Лебедева Г. В. «Программирование для математиков». Эти материалы выгодно отличаются обоснованием построения курса, рассматриваемых авторами тем и задач к ним.

Занятия по информатике в начальной школе изучаются по работам Перевина Ю. А., в частности, по его книге «Методика раннего обучения информатике», в которой автор тоже обосновывает свой выбор тем, выбор целей

и задач курса. Данный материал может быть дополнен разбором занятий с ресурса урокицифры.рф.

В качестве учебника по углубленному курсу информатики в старшей школе изучается комплект учебников Полякова К. Ю. и Еремина Е. А., содержащий разнообразный набор тем и задач различной сложности.

Основной обсуждаемый дидактический подход — это Большая Дидактика Е. Яновицкой и М. Адамского. Этот подход интересен и полезен тем, что его разработали и применяют в различных школах, в том числе и неспециализированных; он может быть адаптирован на большинство школьных предметов (и часть вузовских); его идеи и методы позволяют качественно обучать всех и особенно полезны и эффективны для начала обучения предмету и в случаях слабой подготовки учащихся.

Другой методической книгой является пособие Гина А. А. «Приемы педагогической техники: свобода выбора, открытость, деятельность, обратная связь, идеальность», выдержавшее уже 18 переизданий. Книга содержит наборы дидактических приемов и приемов управления классом, которые можно комбинировать друг с другом, тем самым разнообразя форму проведения уроков разных типов.

Методическую копилку дополняет книга Ершова П. М., Ершовой А. П. и Букатова В. М. «Общение на уроке, или Режиссура поведения учителя», в которой систематизированы виды межличностного взаимодействия и приводятся примеры соответствующих педагогических ситуаций. Дальнейшее овладение этой тематикой возможно на основе других работ её авторов.

Дополнительными источниками, которые можно рекомендовать слушателям данного курса, являются «Школа будущего, построенная вместе с детьми» А. Н. Тубельского, «Школа влияния» С. Соловейчика, «Другая школа: образование — не система, а люди» А. И. Мурашева, а также глава 14 «Об обучении, преподавании и обучении преподаванию» известной книги Д. Пойа «Математическое открытие».

Заключение

В работе представлены краткие характеристики курсов методики преподавания информатики/компьютерных наук, читаемых в некоторых непедагогических вузах для обучающихся по ИТ-специальностям.

Список литературы

- [1] Учебные планы и рабочие программы дисциплин СПбГУ:

<https://spbu.ru/sveden/education>

[2] Учебные планы и рабочие программы дисциплин МГУ:

<http://edu.msu.ru/depts.shtml>

[3] Учебные планы и рабочие программы дисциплин НГУ:

https://www.nsu.ru/n/information-technologies-department/education_fit/programs/

[4] Учебные планы и рабочие программы дисциплин ИТМО:

<https://abit.itmo.ru/master>

[5] Учебные планы и рабочие программы дисциплин ЛЭТИ:

<https://etu.ru/sveden/education/eduop/>

[6] Учебные планы и рабочие программы дисциплин ННГУ:

<http://www.unn.ru/sveden/education/edu-op.php>

[7] Учебные планы и рабочие программы дисциплин КубГУ:

<https://www.kubsu.ru/ru/education/programs>

[8] Учебные планы и рабочие программы дисциплин РГУ:

<https://www.rsu.edu.ru/sveden/education/>

[9] Учебные планы и рабочие программы дисциплин СГУ:

<https://www.sgu.ru/education/courses>

О распараллеливании одного метода бинаризации

Макаров А.А., СПбГУ, Санкт-Петербург a.a.makarov@spbu.ru,
 Савельева М.Ю., СПбГУ, Санкт-Петербург savmary@mail.ru,
 Шабунин А.Н., СПбГУ, Санкт-Петербург shandr52@gmail.com

Аннотация

Анализ создания информационных систем для органов государственной власти и обзор инструментальных средств описания бизнес процессов показал целесообразность создания единой (комбинированной) методологии, сочетающей в себе методологию функционального моделирования IDEF0 и методологию управления бизнес-процессами BPM (подробнее см. [1, 2, 3]). Один из практических аспектов реализации проектов по разработанной методологии заключается в автоматизации документооборота, что позволяет снизить нагрузку на персонал и ведет к уменьшению количества ошибок при оформлении документов [4]. Получаемые электронные документы целесообразно снабжать цифровыми водяными знаками. В качестве цифрового водяного знака может выступать бинарное или бинаризованное полутоновое цифровое изображение [5, 6, 7, 8, 9].

Алгоритмам обработки изображений присущ естественный параллелизм, основанный на декомпозиции данных. Несмотря на наличие различных библиотек, содержащих функции параллельной обработки изображений, использование готовых библиотек имеет ряд ограничений [10]. Ввиду распространения многоядерных процессоров и многопроцессорных систем, учитывая большие временные затраты, необходимые при обработке массивов изображений, требование к увеличению скорости бинаризации остается весьма актуальным.

Список литературы

- [1] Шабунин А.Н. Проектирование и инструментальные средства генерации электронных услуг для органов государственной власти // Тр. СПИ-ИРАН **30** (2013), 301–313.
- [2] Makarov A., Shabunin A. On Design of Secure E-Services for Public Authority in the Russian Federation // 20th Conference of Open Innovations Association (FRUCT), 2017, 260–267.

- [3] Makarov A., Shabunin A. Unified Design Methodology for State Information Systems // Proc. of the V International Workshop on Modeling, Information Processing and Computing (MIP: Computing-V 2022) **3091** (2022), 81–86.
doi:10.47813/dnit-mip5/2022-3091-81-86
- [4] Шабунин А.Н., Макаров А.А. О проектировании информационной системы инспекционно-досмотрового комплекса // Компьютерные инструменты в образовании **2** (2023), 62–78.
- [5] Яковлева Е.С., Макаров А.А. О свойствах блочного алгоритма бинаризации цифровых изображений // Компьютерные инструменты в образовании **4** (2015), 26–36.
- [6] Makarov A., Yakovleva E. Comparative analysis of halftoning algorithms for digital watermarking // 18th Conference of Open Innovations Association and Seminar on Information Security and Protection of Information Technology (FRUCT-ISPIT), 2016, 193–199.
- [7] Макаров А.А., Максимов В.В. О бинаризации гидролокационных изображений // Гидроакустика **30:2** (2017), 42–51.
- [8] Shumilov B., Gerasimova Y., Makarov A. On Binarization of Images at the Pavement Defects Recognition // Proceedings of the 2018 IEEE International Conference on Electrical Engineering and Photonics (EExPolytech 2018), 2018, 107–110.
- [9] Макаров А.А., Савельева М.Ю. О схемах встраивания цифровых водяных знаков в полутоновые изображения // Волновая электроника и инфокоммуникационные системы: XXVII Междунар. науч. конф. (СПб., 3–7 июня 2024 г.): сб. статей: в 3 ч. Ч. 2. – СПб.: ГУАП, 2024, 97–101.
- [10] Верещагин К.А., Макаров А.А. Параллельный алгоритм бинаризации изображения // Процессы управления и устойчивость **3(19)**, no. 1., (2016), 358–361.

Вейвлетные схемы в обработке радиолокационных данных

Косогоров О.М., ГУАП, Санкт-Петербург okosogorov@mail.ru,
 Макаров А.А., СПбГУ, ГУАП, Санкт-Петербург a.a.makarov@spbu.ru,
 Макарова С.В., СПбГУ, Санкт-Петербург sdrobot@mail.ru

Аннотация

Создание теории сплайнов и вейвлетов привело к прорыву в области информационных технологий, в частности к принятию стандарта JPEG 2000. Идея построения вейвлетных схем (например, основанных на декомпозиции пространств сплайнов различного типа) состоит в построении адаптивных сеток, декомпозиции и дальнейшем прореживании данных «по времени» или «по частоте», что приводит к сжатию поступающего цифрового сигнала (подробнее см., например, [1, 2, 3, 4, 5, 6]). Используемые подходы нашли широкое применение при разработке методов обработки потоков цифровой информации, в том числе и радиолокационных данных, что особенно актуально при создании многопозиционных радиолокационных станций [7, 8, 9]. Адаптивные вейвлетные схемы позволяют обеспечить высокий коэффициент сжатия, сохраняя высокую скорость сжатия, и высокую точность восстановления потоков радиолокационной информации при осуществлении передачи данных в режиме реального времени.

Список литературы

- [1] Макаров А.А. О вэйвлетном разложении пространств сплайнов первого порядка // Проблемы матем. анализа, Вып. 38, 2008, 47–60.
- [2] Макаров А.А. Алгоритмы вэйвлетного уточнения пространств сплайнов первого порядка // Тр. СПИИРАН, Вып. 19, 2011, 203–220.
- [3] Макаров А.А. Алгоритмы вэйвлетного сжатия пространств линейных сплайнов // Вестн. С.-Петерб. ун-та. Сер. 1., Вып. 2, 2012, 41–51.
- [4] Демьянович Ю.К. Сплайн-вэйвлеты при однократном локальном укрупнении сетки // Зап. научн. сем. ПОМИ, Т. 405, 2012, 97–118.
- [5] Демьянович Ю.К., Пономарев А.С. О реализации сплайн-всплескового разложения первого порядка // Зап. научн. сем. ПОМИ, Т. 453, 2016, 33–73.

- [6] Макаров А.А. О двух алгоритмах вейвлет-разложения пространств линейных сплайнов // Зап. научн. сем. ПОМИ, Т. 463, 2017, 277–293.
- [7] Косогоров О.М., Макаров А.А., Макарова С.В. О матричном представлении фильтров, соответствующих сплайн-вейвлетам со смещенным носителем // Зап. научн. сем. ПОМИ, Т. 496, 2020, 156–168.
- [8] Косогоров О.М., Макаров А.А. О некоторых проблемах передачи данных в системе, обеспечивающей функционирование многопозиционной РЛС // Метрологическое обеспечение инновационных технологий: V Междунар. форум: сб. ст. / под ред. академика РАН В. В. Окрепилова. — СПб.: ГУАП, 2023. — С. 67–68.
- [9] Косогоров О.М., Макаров А.А. О формировании входного потока данных в системе, осуществляющей цифровую обработку радиолокационного сигнала // Метрологическое обеспечение инновационных технологий: VI Междунар. форум: сб. ст. / под ред. академика РАН В. В. Окрепилова. — СПб.: ГУАП, 2024. — С. 138–139.

Прикладная кибернетика и искусственный интеллект



Кузнецов Николай Владимирович

член-корреспондент РАН, д.ф.-м.н., профессор, заведующий кафедрой
прикладной кибернетики

Анализ разрывной модели фазовой автоподстройки с пилообразной характеристикой фазового детектора

Арсеньев Д.Г., СПбГУ, СПбПУ, Санкт-Петербург d.arseniev@spbu.ru,
 Кузнецов Н.В., СПбГУ, Санкт-Петербург nkuznetsov239@mail.ru,
 Лобачев М.Ю., СПбГУ, Санкт-Петербург st048700@student.spbu.ru

Аннотация

В данной работе в рамках теории систем дифференциальных уравнений с разрывной правой частью рассмотрена задача определения полос удержания, захвата и быстрого захвата для системы управления фазовой синхронизацией (ФАПЧ) с пилообразной характеристикой фазового детектора [1].

Данный доклад является продолжением работы [6], в которой для системы ФАПЧ с идеальным интегрирующим фильтром первого порядка и непрерывной кусочно-линейной характеристикой фазового детектора была аналитически вычислена полоса быстрого захвата. Показано, что формула для полосы быстрого захвата исследуемой системы ФАПЧ с пилообразной характеристикой может быть получена с помощью формулы из [6] подстановкой соответствующего коэффициента в кусочно-линейной характеристике. Заметим, что, например, в работах [2, 3, 4, 5] при нелинейном анализе систем ФАПЧ не обсуждаются описание векторного поля в точках разрыва и возможная неединственность решений.

Список литературы

- [1] Best R., Kuznetsov N., Leonov G. et al. Tutorial on dynamic analysis of the Costas loop // IFAC Annual Reviews in Control. 2016. Vol. 42. P. 27–49.
- [2] Goldstein A. Analysis of the Phase-Controlled Loop with a Sawtooth Comparator // Bell System Technical Journal. 1962. Vol. 41, no. 2. P. 603–633.
- [3] Шахтарин Б. Исследование кусочно-линейной системы ФАП // Радиотехника и электроника. 1969. № 8. С. 1415–1424.
- [4] Protonotarios E. N. Pull-in time in second-order phase-locked loops with a sawtooth comparator // IEEE Transactions on Circuit Theory. 1970. Vol. 17, no. 3. P. 372–378.

- [5] Harb B. A., Al-Ajlouni A., Eyadeh A. A Collocation-Based Algorithm for Analyzing Bifurcations in Phase Locked Loops with Tanlock and Sawtooth Phase Detectors // Mathematical Problems in Engineering. 2018. Vol. 2018. P. 1–7.
- [6] Kuznetsov N., Arseniev D., Blagov M. et al. The Gardner problem and cycle slipping bifurcation for type-2 phase-locked loops // International Journal of Bifurcation and Chaos in Applied Sciences and Engineering. 2022. Vol. 32, no. 9. art. num. 2250138.

Нейронная сеть Хопфилда для решения задач оптимального управления

Антропова Е.Г., СПбГУ, Санкт-Петербург st093661@student.spbu.ru

Аннотация

В данной работе представлено нахождение оптимального управления линейно-квадратичной (linear quadratic, LQ) задачи с использованием непрерывной нейронной сети Хопфилда (continuous Hopfield neural network, CHNN). Описана архитектура нейронной сети, смоделировано решение задачи линейно-квадратичного регулятора при помощи нейронной сети и динамического уравнения Риккати, сделаны выводы.

Введение

Рассматривается непрерывная сеть Хопфилда для решения задачи LQ. Особенности CHNN:

- Обучение: в CHNN обновление нейронов на основе дифференциальных уравнений. При этом минимумы энергетической функции сети и целевой функции задачи оптимизации могут быть сведены к минимуму одновременно;
- Функция активации: используется непрерывная функция активации, часто используется гиперболический тангенс ($f := \tanh$);
- Применение: CHNN часто применяется в задачах, связанных с оптимизацией, ассоциативным запоминанием и других областях, где непрерывные значения большие подходят для моделирования данных.

В данной работе, опираясь на статью [1], исследуется применение нейронной сети CHNN для нахождения оптимального управления задачи LQ. Суть данного подхода состоит в том, что критерий оптимальности задачи LQ преобразуется к виду функции энергии нейронной сети Хопфилда, засчёт этого искомое управление есть выход нейронов.

Постановка задачи линейно-квадратичного управления

Рассматривается дискретный случай задачи оптимального управления с конечным горизонтом.

$$X(k+1) = A(k)X(k) + B(k)U(k) \quad X(0) = X_0 \quad (1)$$

где k – момент времени, $A(k) \in \mathbf{R}^{n \times n}$, $B(k) \in \mathbf{R}^{n \times r}$, $X(k) \in \mathbf{R}^{n \times 1}$ – вектор состояния, $U(k) \in \mathbf{R}^{r \times 1}$ – вектор управления.

Определён критерий оптимальности для задачи LQ:

$$J = X^T(N)H X(N) + \sum_{k=0}^{N-1} \{X^T(k)Q(k)X(k) + U^T(k)R(k)U(k)\} \quad (2)$$

где $H \in \mathbf{R}^{n \times n}$ и $Q(k) \in \mathbf{R}^{n \times n}$, N – горизонт, $0 \leq k < N$ – положительные полуопределённые матрицы. Задача оптимального управления LQ является нахождением последовательности управления $U(0), U(1), \dots, U(N-1)$, минимизирующей критерий оптимальности J .

Решение линейной системы

Для нахождения оптимального управления использована архитектура CHNN, где вектором управления будет выход нейронов нейронной сети.

Записан вектор управления $\tilde{U} \in \mathbf{R}^{rN \times 1}$ для горизонта N :

$$\tilde{U}^T = \{U^T(0), U^T(1), \dots, U^T(k), \dots, U^T(N-1)\},$$

Тогда из уравнения (1) задача перепишется как

$$X(k) = \Phi(k)X(0) + \Psi(k)\tilde{U}, \quad (3)$$

где $\Phi(k) \in \mathbf{R}^{n \times n}$ – матрица передачи состояний системы (1), которая удовлетворяет

$$\Phi(k) = A(k-1)A(k-2)\dots A(0).$$

$\Psi(k) \in \mathbf{R}^{n \times rN}$ находится как

$$\Psi(k) = \begin{cases} \Psi(k) = \{\Psi_1(k), \Psi_2(k), \dots, \Psi_N(k)\} \\ \Psi_i(k) = \begin{cases} A(k-1)A(k-2)\dots A(i)B(i-1) & (i \leq k) \\ 0 & (i > k) \end{cases} \end{cases}$$

Архитектура CHNN

Задано количество нейронов для нейронной сети Хопфилда (CHNN) – $L = rN$, где r – размерность вектора управления, N – временной горизонт.

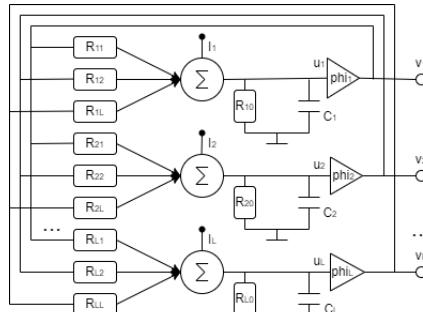


Рис. 1: схема CHNN

Для L -нейронной сети набор дифференциальных уравнений состояний и выходов нейронной сети задаётся формулой

$$C_i \frac{du_i}{dt} = -\frac{u_i}{R_i} + \sum_{j=1}^L w_{ij}v_j + I_i,$$

$$v_i = \varphi_i(u_i), \quad i = 1, 2, \dots, L,$$

где u_{00} – регулируемый параметр, $u_i(t)$ и $v_i(t)$ – вход и выход i -го нейрона в момент t , $\varphi_i(\cdot)$ – функция активации нейросети, I_i – входной ток внешнего смещения, C_i – входная ёмкость нейрона, а R_i – входное сопротивление нейрона, w_{ij} – веса, связывающие i -ый и j -ый нейрон и

$$\frac{1}{R_i} = \frac{1}{R_{i0}} + \sum_{j=1}^L w_{ij}, \quad w_{ij} = \frac{1}{R_{ij}}.$$

Схема непрерывной нейронной сети Хопфилда представлена на Рис. 1.

Хопфилд [2] показал, что при симметричных соединениях ($w_{ij} = w_{ji}$) локальный минимум энергетической функции E , которая определена по формуле (4) достигает устойчивого состояния.

$$E = -\frac{1}{2} \sum_{i=1}^L \sum_{k=1}^L w_{ik} v_i v_k - \sum_{i=1}^L v_i I_i \quad (4)$$

или

$$E = -\frac{1}{2} V^T W V - V^T I,$$

где $V \in \mathbf{R}^{L \times 1}$ – выходной вектор, $W \in \mathbf{R}^{L \times L}$ – матрица весов, $I \in \mathbf{R}^{L \times 1}$ – пороговый вектор. То есть

$$V = (v_1, v_2, \dots, v_L)^T$$

$$I = (I_1, I_2, \dots, I_L)^T$$

$$W = \begin{pmatrix} w_{11} & w_{12} & \dots & w_{1L} \\ w_{21} & w_{22} & \dots & w_{2L} \\ \dots \\ w_{L1} & w_{L2} & \dots & w_{LL} \end{pmatrix}$$

Нахождение решения задачи LQ при помощи СНН

Не умаляя общности, считается $\tilde{U} \in [-1, 1]^{rN \times 1}$, $\tilde{U} = V$, $V = (v_1, v_2, \dots, v_L)^T$, где

$$v_i = \tanh\left(\frac{u_i}{u_{00}}\right), \quad i = 1, 2, \dots, L,$$

Преобразован критерий оптимальности (2) в функцию энергии нейронной сети (4).

$$J = X^T(N)PX(N) + \tilde{U}^T\tilde{R}\tilde{U} + \sum_{k=0}^{N-1} (X^T(k)Q(k)X(k)) \quad (5)$$

где

$$\tilde{R} = \text{diag}(R(0), R(1), \dots, R(N-1)) \in \mathbf{R}^{mN \times mN}$$

Подставлена формула (3) в (5) и получено

$$\begin{aligned} J = & \tilde{U}^T \left(\tilde{R} + \Psi^T(N)P\Psi(N)\tilde{U} + 2X^T(0)\Phi^T(N)P\Psi(N)\tilde{U} \right. \\ & + X^T(0)\Phi^T(N)P\Phi(N)X(0) + X^T(0)QX(0) \\ & \left. + \sum_{k=1}^{N-1} \left\{ \begin{array}{l} \tilde{U}^T\Psi^T(k)Q\Psi(k)\tilde{U} + 2X^T(0)\Phi^T(k)Q\Psi(k)\tilde{U} \\ + X^T(0)\Phi^T(k)Q\Phi(k)X(0) \end{array} \right\} \right) \end{aligned}$$

Тогда критерий оптимальности J эквивалентен

$$\begin{aligned} J_1 &= \tilde{U}^T \left(\tilde{R} + \Psi^T(N)P\Psi(N) \mid \tilde{U} + 2X^T(0)\Phi^T(N)P\Psi(N)\tilde{U} \right. \\ &\quad \left. + \sum_{k=1}^{N-1} \tilde{U}^T \Psi^T(k)Q\Psi(k)\tilde{U} + 2X^T(0)\Phi^T(k)Q\Psi(k)\tilde{U} \right) = \\ &= \tilde{U}^T F \tilde{U} + 2\tilde{U}^T G \end{aligned}$$

где матрицы $F \in \mathbf{R}^{mN \times mN}$ и $G \in \mathbf{R}^{n \times 1}$ рассчитываются как

$$\begin{aligned} F &= \tilde{R} + \Psi^T(N)P\Psi(N) + \sum_{k=1}^{N-1} (\Psi^T(k)Q\Psi(k)), \\ G &= \left(\Psi(N)^T P \Phi(N) + \sum_{k=1}^{N-1} (\Psi^T(k)Q \Phi(k)) \right) x(0). \end{aligned}$$

Тогда матрица весов W и пороговый вектор I находятся как

$$W = -2F, \quad I = -2G.$$

Пример решения задачи оптимального управления

Рассматривается дискретная задача линейно-квадратичного регулятора (linear quadratic regulator, LQR) с конечным горизонтом для проверки работоспособности метода:

$$\begin{cases} X(k+1) = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix} X(k) + \begin{pmatrix} 0 \\ 1 \end{pmatrix} U(k) \\ Y(k) = \begin{pmatrix} 1 & 0 \end{pmatrix} X(k) \end{cases}, \quad (6)$$

где

$$X(0) = \begin{pmatrix} 1 & 0 \end{pmatrix}^T,$$

и весовые матрицы:

$$Q = H = C^T C, \quad R = \rho I, \quad N = 3,$$

I – единичная матрица, $\rho = 0.3$.



Рис. 2: результат моделирования метода с CHNN

CHNN построила оптимальное управление для задачи LQR (6). Нейронная сеть Хопфилда имеет $L = 3$ нейрона и функцию активации – гиперболический тангенс ($f := \tanh$). При реализации использованы такие библиотеки как Numpy, Scipy, Matplotlib. Программа запускалась на персональном компьютере. На Рис. 2 представлены графики выходов нейронов нейросети в зависимости от времени, красными точками обозначены решения, полученные с помощью Риккати.

Заключение

Описано применение нейронной сети Хопфилда к нахождению оптимального управления LQ, представлена архитектура нейронной сети. Смоделирована CHNN для решения дискретной задачи LQR с конечным горизонтом. Найденное с помощью этого метода оптимальное управление для задачи LQR (6) сошлось с управлением, полученным с помощью Риккати. Можно сделать вывод, что данный метод применим для такого типа задач.

Список литературы

- [1] M. Li, X. Ruan Optimal Control with Continuous Hopfield Neural Network // International Conference on Robotics, Intelligent Systems and Signal Processing. Changsha, 2003.
- [2] J.J. Hopfield, D.W. Tank Neural Computation of Decisions in Optimization Problems // Biological Cybernetics. 1985. Vol. 52. P. 141-152.
- [3] M. Lan, S. Chand Solving linear quadratic discrete-time optimal controls using neural network // Proceedings of the 29th Conference on Decision and Control Honolulu. 1990. Hawaii.

Глобальная устойчивость нейронных сетей с недифференцируемыми активационными функциями

Кисиев Т.А., СПбГУ, Санкт-Петербург st085727@student.spbu.ru,
 Мокаев Т.Н., СПбГУ, Санкт-Петербург t.mokaev@spbu.ru

Аннотация

В сообщении представлены результаты, полученные в рамках исследования устойчивости нейронных сетей с негладкими активациями. В процессе исследований были продолжены результаты авторов М. Форти и П. Нистри [9]. В частности, доказаны теоремы, гарантирующие устойчивость нейронной сети Хопфилда для более широкого класса функций активации нейронов.

Введение

Однослойные рекуррентные нейронные сети Хопфилда [5] часто используются для решения оптимизационных задач гладкого линейного и нелинейного программирования, когда целевая функция и ограничения являются непрерывно дифференцируемыми функциями.

Основой для формирования архитектуры сети является метод штрафов, включающий в себя градиентную систему энергетической функции, которая, в свою очередь, складывается из целевой функции и функций из ограничений конкретной оптимизационной задачи. Обусловлено это тем, что по энергетической функции можно задать активационные функции, связи между нейронами и определить входные данные для сети. Вычисление решения заключается в предоставлении неких начальных состояний (напряжений) для нейронов, далее нейронная сеть сойдется к устойчивому состоянию равновесия, которое соответствует минимуму энергетической функции, который, в свою очередь, соответствует минимуму функции затрат. Логично, что для таких систем важно наличие единственного глобально асимптотически устойчивого состояния равновесия.

Необходимым условием для сходимости метода штрафов за конечное время является использование недифференцируемых функций в ограничениях [2]. Эти функции отражены активационными функциями нейронов сети, поэтому возникает вопрос устойчивости сетей с негладкими активациями. В работе М. Форти и П. Нистри [9, см. теоремы 1 и 3] доказано наличие единственного глобально асимптотически устойчивого состояния равновесия у

сети Хопфилда с разрывной, ограниченной и монотонной функцией активации. Можно обобщить эти результаты для неограниченных и немонотонных разрывных функций, тем самым расширяя класс оптимизационных задач, которые можно эффективно и точно решать с помощью таких сетей.

Биологическая и математическая модели сети

Чтобы лучше понять концепции, использованные при создании и описании модели нейронной сети Хопфилда, кратко расскажем о её биологическом прародителе – человеческом мозге.

Как известно из биологии, мозг человека состоит примерно из 10^{12} нейронов. Они суть вычислительные единицы мозга. Между ними есть связующие отростки – аксоны (рис. 1). Через них нейроны передают друг другу нервные импульсы, которые принимаются дендритами. Сигналы передаются за счет разности зарядов на внешней и внутренней сторонах мембранны аксона, которая, в свою очередь определяется разностью в количестве ионов натрия и калия. Следовательно, активация нейрона ассоциируется с распространением импульса электрического напряжения вдоль цилиндрической мембранны аксона. Когда суммарный ток от других нейронов превышает определенный порог, нейрон генерирует собственный импульс, после которого на время становится невосприимчивым к внешним воздействиям (состояние рефрактерности).

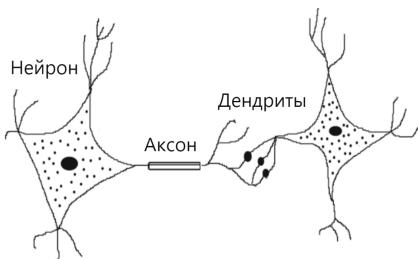


Рис. 1: Биологические нейроны и связи между ними.

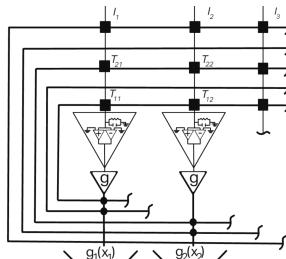


Рис. 2: Электронная цепь, моделирующая нейронную сеть.

Так как биологические нейроны передают друг другу электрические заряды, то аналоговая нейронная сеть, моделирующая абстрактную, может представлять собой электронную цепь Кирхгофа (рис. 2). Такой подход был предложен в 1952г. в работе Ходкина и Хаксли [6].

Цепь состоит из искусственных нейронов, которые, по сути, являются электрическими усилителями. Напряжения на них изменяются с течением

времени, отражая изменения состояний реальных нейронов. Связи между нейронами осуществляются включением в цепь резисторов проводимостью T_{ij} и сопротивлением $R_{ij} = \frac{1}{|T_{ij}|}$. Функция активации $g_i(x_i(t))$ характеризует выходное напряжение на i -ом нейроне относительно поступившего в него напряжения $x_i(t)$. Пауза в восприимчивости нейрона осуществляется за счет конденсатора емкости C_j . Внешние сигналы I_j зависят от постановки конкретной задачи, для решения которой адаптируется сеть.

В системе явно задается динамика, поэтому математически её можно описать как динамическую систему. Для задания векторного поля динамической системы нужны дифференциальные уравнения, вывести которые можно из физических законов. По первому закону Кирхгофа суммарный ток, поступающий в нейрон, равен суммарному току, исходящему из него:

$$\frac{x_j(t)}{R_j} + C_j \frac{dx_j(t)}{dt} = \sum_{i=1}^N T_{ij} g_i(x_i(t)) + I_j. \quad (*)$$

Каждое уравнение типа (*) моделирует изменение с течением времени состояния нейрона в сети. Объединив эти уравнения в систему, получим систему дифференциальных уравнений (1), моделирующую нейронную сеть:

$$\dot{x}(t) = f(x(t)) = Bx(t) + Tg(x(t)) + I, \quad (1)$$

где матрица $B = \text{diag}(-b_1, \dots, -b_n) \in M_n(\mathbb{R})$, $b_i > 0$, – включает в себя емкости и сопротивления, тем самым моделирует рефрактерное состояние нейрона, T – матрица связей нейронов.

Определение решений системы

Функция активации в правой части системы (1) в данной теории является разрывной функцией с конечным числом разрывов первого рода, причем для односторонних предельных значений функции в точках разрыва верно соотношение $g_i(\rho_k^+) > g_i(\rho_k^-)$. В дальнейшем такие функции будем считать принадлежащими классу \mathcal{G}' . Важно подметить, что в отличие от работы [9], тут от функции не требуется монотонность и ограниченность.

Для определения решений системы с разрывной правой частью избрана теория А. Филиппова [4].

Смысл определения *решений по Филиппову* в том, что касательный вектор к траектории решения, где он существует, должен лежать в замыкании выпуклой оболочки⁰ предельных значений векторного поля $f(x(t))$ в сколь

угодно малых окрестностях траектории. Таким образом, система (1) записывается как система дифференциальных включений:

$$\begin{aligned}\dot{x}(t) \in \varphi(f(x(t))) &= \overline{\text{conv}}\{\lim f(x_i(t)) \mid x_i(t) \rightarrow x(t), x_i(t) \notin N_f\} = \\ &= Bx(t) + T \overline{\text{conv}}[g(x(t))] + I,\end{aligned}\quad (2)$$

Важно, что мы исключаем множества меры нуль N_f , где функция имеет разрыв. Это позволяет определить траектории в точках, в которых само векторное поле, вообще говоря, не определено. Используя определение решений по Филиппову, можно также определить *состояние равновесия системы* (2) как определенное на полуинтервале $[0, +\infty)$ стационарное решение дифференциального включения:

$$0 \in \varphi(x^*) = Bx^* + T \overline{\text{conv}}(g(x^*)) + I.$$

Инструменты для анализа устойчивости

Для анализа глобальной асимптотической устойчивости состояния равновесия системы (2) было решено использовать обобщенный подход Ляпунова [8], который включает в себя исследование знакоопределенности производной в силу системы специально подобранный функции Ляпунова типа Лурье-Постникова, которая также является энергетической функцией нейронной сети Хопфилда [5]. С помощью *обобщенного градиента по Кларку* [3] и *правила цепочки для регулярных функций* [10] можно модифицировать *принцип инвариантности ЛаСалля* [7, Следствие 4.2, стр. 129] и применить его для дифференцируемой почти всюду функции Ляпунова.

Устойчивость состояния равновесия нейронной сети

Условия С1.-С3., гарантируют для нейронной сети (1) существование решения (теорема 1.), а также существование (теорема 2.) единственного глобально асимптотически устойчивого состояния равновесия (теорема 3.).

⁰ $\text{conv}(g(x))$ – замыкание выпуклой оболочки предельных значений в точке.

Вывод этих условий является одним из основных результатов аналитической работы, представленных в данном сообщении.

C1. Найдутся постоянные $a, b \geq 0$, такие, что: $\|\overline{\text{conv}}[g(x)]\| \leq a\|x\| + b$.

C2. Найдется постоянная C_i , такая, что для $\forall \eta_{1,2i} \in \|\overline{\text{conv}}[g_i(x_{1,2})]\|$:

$$\eta_{1i} - \eta_{2i} \leq C_i(x_1 - x_2), \quad \forall x_1, x_2 \in \mathbb{R}, \forall i \in 1 : n.$$

C3. Найдется $\alpha = \text{diag}\{\alpha_1, \dots, \alpha_n\}$, $\alpha_i > 0$ такая, что матрица $\alpha T + T^T \alpha$ отрицательно определена, и:

$$4C_i\alpha_i b_M^2 \| -B^{-1}T \|_2^2 < -\lambda_M b_m, \quad \forall i \in 1 : n,$$

$$\text{где } \lambda_M = \rho(\alpha T + T^T \alpha), b_m = \min_{1 \leq i \leq n} b_i, b_M = \max_{1 \leq i \leq n} b_i.$$

Перед тем как исследовать глобальную асимптотическую устойчивость, нужно убедиться в том, что решения системы (1) существуют на промежутке $[0, +\infty)$.

Теорема 1. (о существовании и продолжении решений)

Пусть активационная функция нейронной сети принадлежит расширенному классу $g(x) \in \mathcal{G}'$, и выполнено условие C1., тогда интервал существования каждого решения системы (1) с начальными данными $x(0) = x_0$ равен $[0, +\infty)$.

Замечание. Теорема была доказана с помощью неравенства Громуолла и теорем о существовании и продолжении решений дифференциальных включений А.Ф. Филиппова [4, теоремы 1 и 2, стр. 77-78].

Теорема 2. (о существовании состояния равновесия сети)

Пусть $g \in \mathcal{G}'$ и выполнены условия C1.-C3., тогда у нейронной сети (1) существует состояние равновесия.

Замечание. Доказательство данной теоремы было проведено путем применения результатов теоремы Лере-Шаудера о неподвижной точке [1, стр.90 т.3.2.8]. В статье М. Форти и П. Ниэтри [9] схожий результат доказывался с помощью теоремы Каутани о неподвижной точке [1, стр. 87, т. 3.2.3]. Однако использование этой теоремы более не даст желаемого результата, так как для её применения была необходима ограниченность функции активации. Поэтому было получено новое доказательство.

¹ $\|A\|_2 = \sqrt{\rho(A^T A)}$, где ρ – спектральный радиус.

Теорема 3. (о глобальной асимптотической устойчивости единственного состояния равновесия сети)

Пусть функция активации $g \in \mathcal{G}'$ и выполнены условия С1.-С3., тогда для любого входного $I \in \mathbb{R}^n$ у сети (1) существует единственное глобально асимптотически устойчивое состояние равновесия.

Замечание. Теорема доказана с помощью принципа инвариантности Ласалля, модифицированного для регулярных функций Ляпунова [10]. Для определения производной в силу системы функции Ляпунова использовалось понятие обобщенного градиента по Кларку и теорема о правиле цепочки для регулярных функций.

Пример

Подкрепим аналитические результаты теоремы 3. примером. Рассмотрим систему дифференциальных уравнений:

$$\begin{cases} \frac{dx_1}{dt} = -x_1 - \frac{1}{4}f_1(x_1) - \frac{1}{8}f_2(x_2), \\ \frac{dx_2}{dt} = -x_2 + \frac{1}{8}f_1(x_1) - \frac{1}{4}f_2(x_2). \end{cases} \quad (1')$$

Где в качестве функции активации возьмем:

$$f_i(x_i) = \tanh(x_i) - \text{sign}(x_i)x_i + \text{sign}(x_i), \quad i \in 1 : 2.$$

Она имеет разрыв первого рода и не является ограниченной и монотонной (рис. 3). Условия теоремы 3. выполнены, значит, в данном случае начало координат $(0, 0)$ есть глобально асимптотически устойчивое состояние равновесия, что видно по фазовому портрету (рис. 4).

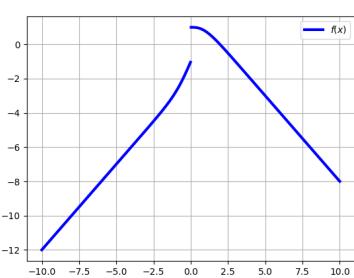


Рис. 3: График функции активации $f(x)$ для сети (1), пример

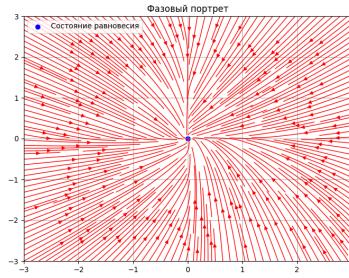


Рис. 4: Фазовый портрет системы ДУ из примера 1.

Заключение.

В процессе работы над материалами сообщения был избран подход для определения решений системы дифференциальных уравнений с разрывной правой частью. Также подобраны теоремы и инструменты для анализа устойчивости негладких систем, на базе которых были выведены и доказаны теоремы, являющиеся обобщением результатов из [9, теоремы 1 и 3] на случай, когда функция активации не является монотонной и ограниченной.

Список литературы

- [1] J. P. Aubin. Set-valued analysis. 1990.
- [2] D. P. Bertsekas. Necessary and sufficient conditions for a penalty method to be exact. *Mathematical Programming*, 9:87–99, 1975.
- [3] F. H. Clarke. Optimization and nonsmooth analysis. 1983.
- [4] A. F. Filippov. Differential equations with discontinuous righthand sides. In *Mathematics and Its Applications*, 1988.
- [5] J. J. Hopfield. Neurons with graded response have collective computational properties like those of two-state neurons. *Proceedings of the National Academy of Sciences of the United States of America*, 81 10:3088–92, 1984.
- [6] A. Hodgkin, A. Huxley. A quantitative description of membrane current and its application to conduction and excitation in nerve. *Bulletin of Mathematical Biology*, 52:25–71, 1952.
- [7] H. K. Khalil. Nonlinear systems third edition. 1992.
- [8] A. M. Lyapunov. The general problem of the stability of motion. 1892.
- [9] M. Forti, P. Nistri. Global convergence of neural networks with discontinuous neuron activations. *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, 50(11):1421–1435, 2003.
- [10] D. W. Shevitz, B. Paden. Lyapunov stability theory of nonsmooth systems. *Proceedings of 32nd IEEE Conference on Decision and Control*, pages 416–421 vol.1, 1993.

Автоматический контроль качества конвейеров данных

Королев А.С., СПбГУ, Санкт-Петербург st087375@student.spbu.ru,
Благов М.В., СПбГУ, Санкт-Петербург m.blagov@spbu.ru

Аннотация

В рамках данной статьи было разработано приложение для автоматического контроля качества данных, состоящее из двух модулей – генератора данных и программы для проверки конвейера. Приложение реализовано на языке программирования Python с использованием библиотек Faker, Pandas, Pyspark. Приложение валидирует запрашиваемые данные, использует разработанный алгоритм генерации датасетов и сравнивает результаты своей работы с работой конвейера данных. Приложение позволяет тестировать конвейеры данных, которые работают на основе операций: left join, right join, outer join, inner join.

Введение

Конвейер данных – приложение, используемое для передачи данных. Большинство современных ИТ компаний используют конвейеры данных для доставки данных в аналитическое хранилище с целью использования этой информации для улучшения своих продуктов и сервисов[4]. Как и любое другое приложение конвейеров данных необходимо тестировать, но тестирование конвейеров данных отличается от тестирования других приложений. Конвейеры данных представляют собой интеграционные пайплайны, на входе и на выходе у которых датасеты. Из чего вытекают следующие проблемы:

- Необходимо тестировать входные данные
- Разные части конвейера данных реализованы при помощи разных инструментов и все нужно протестировать
- Трудоемкость подготовки тестовых данных

Среди перечисленных проблем первые две уже имеют решения и существуют специальные профессии, в сферу обязанностей которых входит решение данных трудностей[1].

Последняя проблема на текущий момент не имеет готового решения и является ключевой для данной работы[5]. Дополнительно отметим, что для тестирования нельзя использовать данные из продуктовой среды из-за политики обработки чувствительных данных.

Постановка задачи

Конвейер данных

Математически конвейер данных может быть описан как отображение T , определенное на множестве входных значений в множество выходных значений. В нашем случае множество входных значений представляет собой кортеж датасетов D_1, \dots, D_n , а множество выходных значений - датасет D' . Датасетом мы будем называть набор колонок и строк $D_i = (r_k, c_j)_{k,j=1}^n$, где каждая колонка может являться одним из следующих типов данных: целое значение, значение с плавающей точкой, дата, булевское значение, строки произвольной длины.

$$T : D_1 \times D_2 \times D_3 \times \dots \times D_n \mapsto D' \quad (1)$$

В течение работы конвейера над изначальными датасетами производится набор операций-преобразований. Обозначим за T_i преобразование на i -том этапе работы конвейера данных.

$$T_i : (D_1, D_2, \dots, D_n) \mapsto D' \quad (2)$$

После i -го этапа выходные данные передаются на следующий этап и конвейер работает с этими данными как с входными. Если обозначить считывание данных как R , а запись как W , которые можно определить как:

$$R : D_1 \times D_2 \times D_3 \times \dots \times D_n \mapsto (D_1, D_2, \dots, D_n) \quad (3)$$

$$W : (D'_1, D'_2, \dots, D'_n) \mapsto D' \quad (4)$$

Благодаря введенным определениям. конвейер данных можно представить как композицию отображений:

$$T(D_1 \times D_2 \times D_3 \times \dots \times D_n) = W(T_n(T_{n-1}(\dots(T_1(R(D_1 \times D_2 \times D_3 \times \dots \times D_n)))))) \quad (5)$$

Также для итоговых датасетов введем определение эквивалентности 2 датасетов.

Определение: 2 датасета эквивалентны (\sim), если датасеты содержат одинаковые записи с точностью до перестановки строк и столбцов.

Генерация датасетов

Чтобы протестировать конвейер данных, нам необходимо заранее будет подготовить датасеты. Для подготовки n датасетов мы требуем на вход n соответствующих схем, $n - 1$ связь между датасетами, $n - 1$ условие соединения и последовательность размером n ключей, которые должны быть одинаковы в следующих друг за другом датасетах.

Таким образом задача тестирования конвейеров данных формулируется так: необходимо генерировать такой набор датасетов D_1, \dots, D_n, D' , чтобы при известном T выполнялось равенство:

$$T(D_1, \dots, D_n) = D' \quad (6)$$

Поставив задачу, перейдем к алгоритму генерации датасетов, который позволит подготовить тестовые данные для конвейера данных.

Генерация удовлетворительных данных

Алгоритм генерации датасетов: генерация произвольного D_i

Прежде всего договоримся, что на вход в нашу программу будет подаваться конфигурационный файл с описанием датасетов. Прежде чем описать требования к конфигурационному файлу, введем определение схемы.

Определение: Схемой датасета называется список столбцов этого набора данных с описанными типами данных и их ограничениями.

От конфигурационного файла мы будем требовать:

1. Количество датасетов и соответствующие им схемы
2. Размеры датасетов и условиях их соединения
3. Количество коррелирующих ключей в датасетах
4. Ограничения для каждого столбца в датасете или список возможных значений для конкретной колонки в датасете

Конфигурационный файл будем называть корректным, если выполняется следующее

1. Количество датасетов равно количеству коррелирующих ключей - 1 и равно количеству соединений - 1

2. Для соединения используются только операции inner-join, outer-join,right-join,left-join
3. Последовательность коррелирующих ключей является невозрастающей
4. Ограничения на столбцы позволяют сгенерировать столько значений, сколько требуется размерами датасета

В дальнейшем будем отталкиваться от того, что пользователь ввел корректную конфигурацию. В качестве T_i рассматриваем соединение по одному или нескольким полям.

Алгоритм генерации датасетов: генерация D_1, D_2 (база)

Сперва возьмем 2 схемы для датасетов A_1, A_2 с размерами n, m соответственно. Затем сперва сгенерируем k_1 ключей соединения, которые будут не совпадать между собой, но будут присутствовать в обоих датасетах:

$$\forall i \neq j, i \in \{1 : k_1\}, j \in \{1 : k_1\} : x_i \neq x_j, \forall x_i, x_j \in \{x_1, x_2, \dots, x_{k_1}\} \quad (7)$$

Затем генерируем оставшиеся $n - k_1, m - k_1$ ключей так, чтобы они не пересекались между собой и не совпадали с исходными k_1 ключами соединения. После того, как все ключи соединения были сгенерированы, генерируем данные для оставшихся столбцов в обоих датасетах и выполняем операцию соединения, заданную пользователем и получаем датасет $A_{12} = A_1 \text{ join } A_2$, длина которого равняется k_1 в случае inner-join.

Алгоритм генерации датасетов: генерация D_n, D_{n+1} (переход)

Теперь нам необходимо сгенерировать датасет A_3 размером l с k_2 ключами соединений, имея на руках датасет A_{12} . Для того, чтобы это сделать скопируем первые k_2 значений из датасета A_{12} и вставим в соответствующие столбцы. Для столбцов, которые отсутствовали в таблице A_{12} будем генерировать новые значения.

После копирования генерируем оставшиеся $l - k_2$ ключей соединения так, чтобы они не пересекались с исходными k_2 ключами и между собой. Затем также генерируем оставшиеся значения для датасета A_3 и объединяем его с датасетом A_{12} по соединению, которое задал пользователь и получаем датасет $A_{123} = A_{12} \text{ join } A_3$

Алгоритм генерации датасетов: результат

После выполнения приложения у нас будут датасеты: A_1, \dots, A_n , сохраненные в формате CSV или Parquet (в зависимости от выбора пользователя) и итоговый датасет $A_{123\dots n}$

Реализация

Входные данные

На вход в программу пользователь передает yaml-файл с конфигурацией для генерации датасетов, путь до конвейера данных, путь до получившихся датасетов и путь для сохранения результата для spark приложения. На выходе пользователь получает ответ относительно результата работы конвейера: получили ожидаемый результат или неожидаемый и надо исправлять код конвейера. Пример конфигурационного файла[2]

Структура приложения

Приложение состоит из 3 классов: config_manager,sample_generator,tester.

Класс config_manager

Класс config_manager парсит схему и валидирует ее. При валидации запускаются набор тестов, которые проверяют ее на валидность, а именно:

- Проверка на количество датасетов, соединений и количество коррелирующих ключей и размеры датасетов. Между ними должно быть следующее соответствие: количество схем датасетов = количество соединений + 1, количество схем датасетов = количество коррелирующих ключей + 1, а также последовательность коррелирующих ключей должна быть не строго убывающей, а также размеры датасетов должны быть не меньше количества требуемых коррелирующих значений.
- Проверка на то, что общие ограничения, уникальные ограничения и списки возможных значений соответствуют размерам датасета.
- Проверка на то, что коррелирующие поля обладают одним типом данных

- Проверка на то, что при всевозможных различных ограничениях на коррелирующие столбцы количество возможных ключей удовлетворяет размерам датасета

В случае, если хотя бы 1 тест не прошел, поднимается ошибка с сообщением о необходимости исправить исходный конфигурационный файл.

Класс sample_generator

Этот класс реализует алгоритм генерации датасетов, который описан выше, а также агрегирует при заданном условии в конфигурации датасеты и сохраняет их в формате csv или parquet в указанной директории.

Класс tester

Класс tester - класс-тестировщик, который на вход принимает путь до конфигурационного файла, пути, где будут лежать итоговые датасеты для конвейера данных, путь до конвейера данных и путь, куда сохранить результат работы конвейера.

Сперва класс вызывает класс *config_manager*, который парсит конфигурационный файл и валидирует его, а затем передает все данные в класс *sample_generator* и происходит генерация датасетов. После окончания генерации, запускается конвейер данных с сгенерированными датасетами, а затем берутся результаты работы генератора датасетов и конвейера данных и сравниваются с учетом введенного отношения эквивалентности. Если конвейер данных выдает неожидаемый результат, то поднимается ошибка, иначе выводится сообщение об успешной работе конвейера данных.

Полный код приложения можно найти в [3]

Заключение

В рамках данной работы было проведено исследование существующих инструментов и ничего не было найдено на тему автоматического контроля качества конвейеров данных. Был разработан и реализован алгоритм генерации датасетов, на основе которого было разработано CLI-приложение на языке программирования Python с использованием библиотек Faker, Pandas, PySpark для автоматического контроля качества конвейеров данных.

Список литературы

- [1] Кто такой и чем занимается Data QA Engineer, 2021. URL: <https://habr.com/ru/companies/skillfactory/articles/591061/>
- [2] Пример конфигурационного yaml-файла, 2024 URL: <https://github.com/prostotema1/Graduation-work/blob/master/config.yaml>
- [3] Код приложения, 2024. URL: <https://github.com/prostotema1/Graduation-work>
- [4] Что такое конвейер данных? И почему вы должны это знать, 2023. URL: <https://habr.com/ru/sandbox/188820/>
- [5] Automating Data Quality Check in Data Pipelines,2023. URL:<https://dzone.com/articles/automating-data-quality-check-in-data-pipelines>
- [6] Faker documentation,2024. URL:<https://faker.readthedocs.io/en/master/>
- [7] PySpark documentation,2024.
URL: <https://spark.apache.org/docs/latest/api/python/index.html>

Telegram-бот для обработки стеганографии в PNG-файлах

Латанов К.В., СГУ, Саратов latanov.kirill@yandex.ru,

Благов М.В., СПбГУ, Санкт-Петербург m.blagov@spbu.ru,

Кузнецов Н.В., СПбГУ, Санкт-Петербург n.v.kuznetsov@spbu.ru

Аннотация

В статье рассматривается метод стеганографического кодирования информации в изображениях формата PNG (Portable Network Graphics), который позволяет встраивать секретные текстовые данные в цифровые изображения без видимого ухудшения их качества и потери целостности. В работе описан алгоритм встраивания и извлечения информации при помощи метода наименее значащих бит (LSB), и Telegram-бот, разработанный с использованием языка программирования Python. Помимо основных стеганографических методов, для усиления защиты бот предоставляет три криптографических метода – шифр Виженера, «Кузнецик» и DES.

Введение

Стеганография необходима для конфиденциальной передачи информации, поскольку она позволяет встраивать скрытые данные в безобидные на первый взгляд файлы, такие как изображения, аудио или видео [1]. Эта техника скрывает сам факт передачи секретной информации, что делает ее идеальным инструментом для обхода систем мониторинга. Стеганографические методы также обеспечивают защиту данных от перехвата, что делает их незаменимыми в условиях современных угроз информационной безопасности. Наличие криптографического шифра не всегда гарантирует защиту, а зашифрованное сообщение нередко выглядит неестественно и привлекает внимание. Поэтому совокупность средств и методов стеганографии и криптографии может обеспечить более надёжный канал коммуникации, который для прочтения сообщения необходимо не только взломать, но и обнаружить сам факт передачи [2].

Теоретическая часть и описание бота

Поскольку внедрение информации будет происходить в нефизический объект, то речь в данной статье пойдёт о цифровой стеганографии.

Цифровая стеганография – ответвление стеганографии, суть которой заключается во встраивании добавочной информации в цифровые объекты. Основными принципами цифровой стеганографии являются изменение объекта без утраты возможности их функционирования и неспособность при помощи органов чувств заметить незначительные искажения в сравнении с оригиналом [3].

В этой работе в качестве цифрового контейнера используется изображение в формате PNG. Выбор данного формата обусловлен рядом преимуществ последнего: широкое использование, поддержка различных глубин цвета, поддержка прозрачности, сжатие без потерь, поддержка метаданных [4].

В качестве усиления защиты используются криптографические средства – шифры Виженера, «Кузнецик» [5] и DES [6]. Первый метод выбран в качестве демонстрационного, поскольку прост в реализации и легок в понимании – шифр Виженера является модификацией шифра Цезаря, который, в свою очередь, является моноалфавитным шифром простой подстановки, где каждая буква исходного сообщения, согласно определённому правилу сдвига, заменяется другой буквой этого же алфавита. Два других метода криптографической защиты являются более надёжными и относятся к блочным симметричным шифрам, каждый из которых в своём алгоритме содержит раундовые преобразования.

В качестве платформы для реализации был выбран Telegram, поскольку большинство коммуникаций в современных реалиях проходят в мессенджерах, а данный представитель является одним из самых популярных [7] и обладает удобными средствами для разработки [8].

Техническая реализация

Описание функционала и демонстрация работы бота

Телеграм-бот написан на языке Python при помощи библиотеки aiogram [9]. Поскольку мессенджер уже имеет свой интерфейс, то дополнительная разработка по части UI не требуется, достаточно только добавления обработчика ввода и сообщений.

При запуске бота пользователю предоставляется выбор – выполнять внедрение или извлечение сообщения из контейнера. После этого выбирается способ шифрования (если текст шифровать не требуется, достаточно выбрать поле «Не шифровать» в контекстном меню) (см. Рис. 1), и в случае выбора такового запрашиваются необходимые для дальнейшего алгоритма параметры, такие как открытые ключи, сдвиг и т. п. На последней стадии

остаётся загрузить контейнер для внедрения, который обязательно должен быть изображением в формате PNG (см. Рис. 2). На случай некорректного ввода на каком-то из шагов предусмотрена обработка состояния с возвращением на предыдущий шаг.



Рис. 1: Процесс подготовки сообщения к внедрению.



Рис. 2: Процесс внедрения сообщения в изображение.

Процесс извлечения сообщения из контейнера с опциональным расшифрованием выполняется в обратной последовательности – пользователем в меню выбирается опция извлечения, после чего присыпается контейнер с встроенным в него сообщением (см. Рис. 3). Далее выбирается поле шифра (если шифрование не применялось, то необходимо указать «Без шифра»), и при необходимости запрашиваются требуемые параметры. В результате на по-

следнем шаге работы программы пользователю выводится исходный текст сообщения, внедрённого в контейнер (см. Рис. 4).

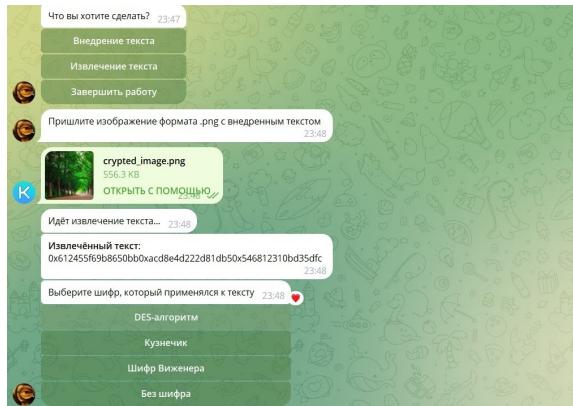


Рис. 3: Процесс получения сообщения из изображения.

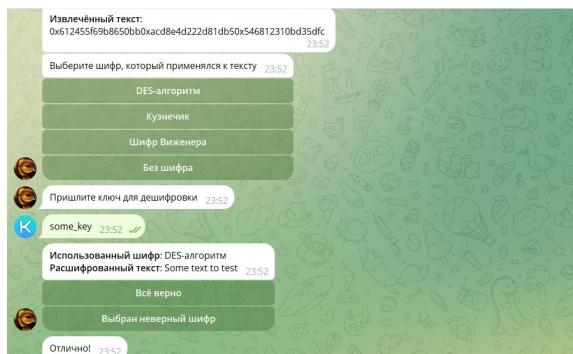


Рис. 4: Процесс получения исходного текста из контейнера.

Заключение

В данной работе была рассмотрена стеганосистема, состоящая из контейнера с изображением в формате PNG, в который встраивается текстовое сообщение. Путём дополнительной обработки перед внедрением можно усил-

лить защиту канала связи, воспользовавшись одним из трёх методов криптографической защиты – шифром Виженера, «Кузнечик» или DES [10].

В качестве платформы для реализации стеганосистемы был выбран Telegram, поскольку данный мессенджер является весьма востребованным, имеет удобный инструментарий для взаимодействия и разработки, а также привычный пользователю интерфейс.

Список литературы

- [1] Семененко, В. А. Информационная безопасность: учебное пособие для вузов / В. А. Семененко. – 2-е изд. – Москва: МГИУ, 2005. – 215 с.
- [2] Рябко Б.Я. Основы современной криптографии и стеганографии. М.: Горячая линия – Телеком, 2013. 232 с
- [3] Грибуин В.Г. Цифровая стеганография. М.: «Солон-пресс», 2020. 262 с
- [4] PNG: особенности формата, преимущества и применение в компьютерной графике <https://nauchniestati.ru/spravka/png>
- [5] Бабенко Л.К., Ищукова Е.А., Толоманенко Е.А. Дифференциальный анализ шифра «Кузнечик». Известия ЮФУ. Технические науки. 2017 г.
- [6] Paar C., Pelzl J. Understanding Cryptography: A Textbook for Students and Practitioners . Springer, 2009. 390 с.
- [7] Популярность мессенджеров в 2024-м: как они будут развиваться <https://mindbox.ru/journal/education/populyarnye-messendzheri/>
- [8] 14 best Python Telegram Bot libraries in 2024 <https://kandi.openweaver.com/collections/python/python-telegram-bot>
- [9] Знакомство с aiogram <https://mastergroosha.github.io/aiogram-3-guide/quickstart/>
- [10] Латанов К.В. Стеганография в графических файлах. Выпускная квалификационная работа, 2024, Саратовский государственный университет, 2024.

Анализ инструментов быстрого захвата изменений данных

Мажара Е.Н., СПбГУ, Санкт-Петербург st087790@student.spbu.ru

Аннотация

В данной работе рассматривается задача быстрой репликации данных между хранилищем и аналитической системой посредством принципа захвата изменений (Change Data Capture, CDC). Для решения этой задачи рассмотрены инструменты Debezium и Airbyte, разработаны примеры конвейеров данных с их использованием. Для разработанных примеров проведена оценка функциональных и нефункциональных характеристик, таких как возможность захвата изменений разных типов и скорость доставки данных.

Введение

В современном мире хранение и обработка данных из различных источников стала важнейшей задачей практически в любой технологической сфере. Современные приложения часто используют несколько хранилищ данных для обслуживания различных задач: хранения информации и её анализа. Используются специализированные аналитические хранилища, способные повысить эффективность анализа данных и снизить нагрузку на основное хранилище [1].

Поскольку данные в хранилище-источнике могут меняться: добавляться, изменяться или удаляться – возникает проблема согласования данных между хранилищем-источником и аналитическим хранилищем. Согласно исследованию [2], принцип полного обновления хранилища для получения изменений неэффективен. Поэтому для решения проблемы синхронизации данных применяется инкрементальный паттерн **CDC** (*Change Data Capture, Захват изменений данных*), используемого для определения изменяемых данных, чтобы предпринять действия с использованием этих данных [3].

В данной работе будет проведен анализ существующих CDC-решений, выбраны наиболее подходящие кандидаты и на их основе построены конвейеры данных, решающие задачу репликации в аналитическое хранилище. Конвейеры затем будут протестированы на предмет скорости доставки изменений.

Анализ существующих решений

На данный момент на рынке представлено более десятка инструментов с необходимой функциональностью. В рамках работы была проанализирована документация и исходный код десяти CDC-инструментов, которые затем были отфильтрованы следующими критериями:

- Открытость исходного кода инструмента;
- Дата последнего обновления либо минорного изменения;
- Популярность и масштаб использования (на основе информации из репозитория с кодом приложения);
- Возможность отслеживания изменений данных в базе **PostgreSQL**;
- Публикация изменений в брокер сообщений **Apache Kafka**;
- Поддержка DML-операций **INSERT, UPDATE, DELETE**;
- Реализация *at-least-once* семантики.

В результате были выбраны две платформы: Debezium и Airbyte.

Debezium

Один из самых распространенных способов доставки изменений данных на сегодняшний день. Проект разрабатывается как полностью бесплатный open-source под лицензией Apache License v2.0 и спонсируется Red Hat. Отслеживание изменений происходит методом чтения журнала транзакций (WAL-журнал) [4].

№	Преимущества	Недостатки
1	Виртуализация	Вероятность потери данных
2	Широкий выбор базы-источника	Возможны дубликаты
3	Единая структура сообщений	Снапшот блокирует таблицы

Таблица 1: Особенности Debezium.

Работа Debezium построена на нескольких компонентах: сервер конфигурации Zookeeper, брокер сообщений Kafka и основной – Kafka Connect. Развёртывание кластера Kafka Connect экспонирует REST API, которое позволяет получать информацию о существующих соединениях и создавать новые.

Для создания посылается HTTP POST-запроса с телом, содержащим информацию о создаваемом соединении: параметры БД-источника, список отслеживаемых таблиц. Изменения приходят в брокер в формате JSON и содержат данные о типе операции, моменте её фиксации, схеме и изменениях данных.

Airbyte

Платформа Airbyte предоставляет более широкую функциональность: это целая система хранения и обработки данных, которая предоставляет услуги облачного хранилища, а также позволяет настраивать соединения между различными источниками и получателями данных. Проект разрабатывается как частично бесплатный с открытым исходным кодом под лицензией Elastic License v2.0, его разработка поддерживается платными облачными функциями. Имеется возможность выбора метода отслеживания изменений, в частности, может быть выбран метод чтения журнала транзакций (WAL-журнал) [5].

№	Преимущества	Недостатки
1	Виртуализация	Ручная синхронизация
2	Выбор источника и приёмника	Возможны дубликаты
3	Удобный веб-интерфейс	Система логирования

Таблица 2: Особенности Airbyte.

Кластер Airbyte включает множество компонентов, которые запускаются совместно при помощи единого скрипта; в частности, создаётся локальный сервер, позволяющего управлять соединениями при помощи удобного интерфейса. Для создания CDC-соединения между хранилищем и брокером сообщений необходимо выбрать и настроить источник и приёмник, передав необходимые конфигурационные параметры, такие как JDBC-адрес базы данных и адрес брокера сообщений, название топика для записи, наименования таблиц для отслеживания и т.д., через веб-формы вместо файла. Сообщения также имеют формат JSON и содержат данные о моменте фиксации операции, а также об изменённых данных.

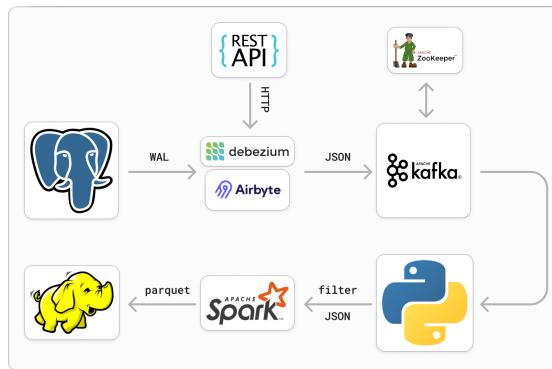


Рис. 1: Схема работы конвейеров данных.

Реализация конвейера данных

Источник

Источником, из которого происходит вычитывание данных для нотификации об изменениях, служит база данных на основе СУБД PostgreSQL [6]. Для отслеживания изменения данных используется одна таблица с несложной схемой данных (4 столбца). Для удобства сервер PostgreSQL с базой запущены в docker-контейнере.

Аналитическое хранилище

В качестве целевого аналитического хранилища используется распределённая файловая система HDFS (Hadoop Distributed File System) [7]. Изменения вносятся в неё в колоночном формате parquet. Запуск кластера HDFS происходит в docker-контейнерах, создаётся одна Datanode.

Доставка изменений

Попавшие в Kafka сообщения вычитываются и фильтруются python-скриптом, а трансформированная запись затем записывается в аналитическое хранилище при помощи Apache Spark [8], который преобразует JSON-объект в parquet-файл [9].

Тестирование

Методика тестирования

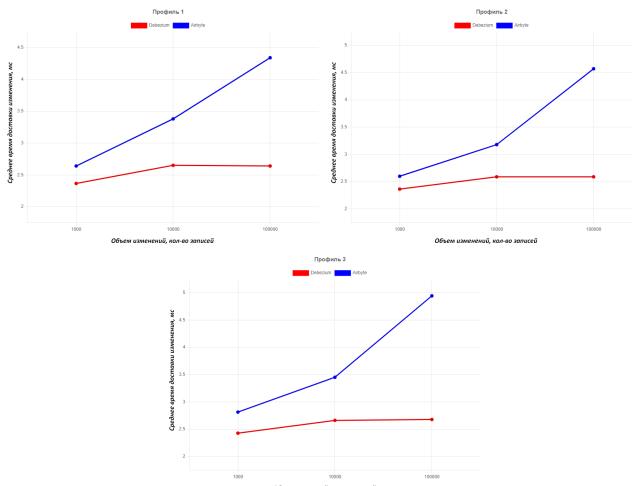
При тестировании измерялось среднее время доставки изменения при 1000, 10000 и 100000 изменяемых записях на трёх профилях, представленных в таблице 3.

№	Профиль 1	Профиль 2	Профиль 3
INS:UPD:DEL	1:1:1	1:0:0	1:1:8

Таблица 3: Профили тестирования.

Для снижения погрешности каждый замер проводился трижды на различных данных. Для вычисления среднего времени использовалась информация о времени доставки последнего сообщения, времени начала транзакции; получаемый интервал делился на количество изменений. Данные изменялись при помощи случайно генерированных python-программой SQL-скриптов.

Результаты



Заключение

В работе был проведён анализ и отбор CDC решений, в результате были выбраны инструменты Debezium и Airbyte. На их основе были созданы конвейеры данных, передающие изменения на источнике в целевое хранилище, которые были протестированы на предмет скорости доставки сообщений на нескольких профилях тестирования.

Результаты показали, что Debezium более стабилен при увеличении количества изменяемых записей. При этом оба инструмента показали схожие результаты на различных профилях, что говорит о несущественной зависимости времени доставки и типа отслеживаемой операции.

Список литературы

- [1] Зайцев Г. Оптимизация применения CRUD операций для аналитического хранилища данных. — 2022. — 41C.
- [2] Jorg T., Dessloch S. Formalizing ETL jobs for incremental loading of data warehouses // Proceedings der 13. GI-Fachtagung f ur Datenbanksysteme in Business, Technologie und Web. Lecture Notes in Informatics. — 2009. — C. 327–346.
- [3] Imani F., Widyasari Y., Arifin S. Optimizing Extract, Transform, and Load Process Using Change Data Capture // 6th International Seminar on Research of Information Technology and Intelligent Systems (ISRITI). — 2023. — C. 266–269.
- [4] Debezium Documentation // URL: <https://debezium.io/documentation/reference/stable/index.html>
- [5] Airbyte Documentation // URL: <https://airbyte.com/>
- [6] PostgreSQL Documentation // URL: <https://www.postgresql.org/>
- [7] Shvachko K., Kuang H., Radia S., Chansler R. The Hadoop Distributed File System // IEEE 26th Symposium on Mass Storage Systems and Technologies (MSST). — 2010. — C. 1–10.
- [8] Spark Documentation // URL: <https://spark.apache.org/>
- [9] Vohra D. Apache Parquet // Practical Hadoop Ecosystem: Apress, Berkeley, CA. — 2016. — C. 325–335.

Изучение динамики клеточных автоматов с помощью нейронных сетей

Мазяр В. А., СПбГУ, Санкт-Петербург st087826@student.spbu.ru,
 Мокаев Т.Н., СПбГУ, Санкт-Петербург t.mokaev@spbu.ru

Аннотация

Исследование динамики клеточных автоматов с помощью нейронных сетей представляет собой актуальную область исследований. Данная работа посвящена изучению возможностей применения нейронных сетей для прогнозирования динамики клеточных автоматов. В работе, на примере известного клеточного автомата – игры «Жизнь», будут рассмотрены и сравнены между собой некоторые методы обучения с учителем, а также предложен и проанализирован способ их ускорения.

Введение

Клеточные автоматы [1] – дискретная модель, основанная на пространстве, состоящем из смежных клеток. Каждая клетка может находиться в одном из конечного числа состояний $\sigma \in \Sigma \equiv \{0, 1, 2, \dots, k - 1, k\}$. Модель может быть бесконечной, конечной закольцованной и конечной замкнутой. Каждая клетка имеет свою окрестность. Устанавливаются правила, которые определяют переходы между состояниями клеток. Один шаг автомата – обход всех клеток, определяющий новое состояние $\sigma(t + 1)$ на основе о текущем состоянии клетки $\sigma(t)$ и ее окрестности $\mathcal{N}(t)$ согласно правилам φ : $\sigma(t + 1) = \varphi(\sigma(t), \mathcal{N}(t))$.

В математическом моделировании неизвестных систем, так называемых «черных ящиков», применяются различные модели, и в том числе клеточные автоматы, для изучения и предсказания поведения таких систем. Клеточные автоматы находят свое применение в криптографии, биологии и моделировании физических, экономических, популяционных процессах. Например, с помощью клеточных автоматов можно моделировать процесс роста кристаллов [2].

Описание модели

Игра «Жизнь»[3] — клеточный автомат, придуманный английским математиком Джоном Конвеем. Она имеет следующие правила:

- Игрок только размещает или случайно генерирует нулевое поколение;

- Каждая клетка на этой поверхности имеет восемь соседей, окружающих ее, и может быть живой или мертвый;
- Каждое следующее поколение рассчитывается на основе предыдущего с использованием следующих правил:
 - если мертвая клетка имеет три живые соседние клетки, то там зарождается жизнь;
 - если живая клетка имеет две или три живые соседние клетки, то эта клетка продолжает жить. В противном случае, клетка умирает.
- Игра прекращается, если
 - конфигурация на очередном шаге точно повторяет одну из более ранних конфигураций;
 - на очередном шаге ни одна из клеток не меняет свое состояние.

Многослойный перцептрон

Рассмотрим перцептрон с двумя входными узлами (состояние клетки и количество соседей у нее) и одним выходным (следующее состояние). Потребуется один скрытый слой [4], так как аппроксимация игры «Жизнь» не настолько сложная задача, и два скрытых узла в нем по первому пункту правил Хитона [4]. Получается перцептрон (рис 1.) с шестью весами w_i и тремя порогами b_i , которые нужно настроить.

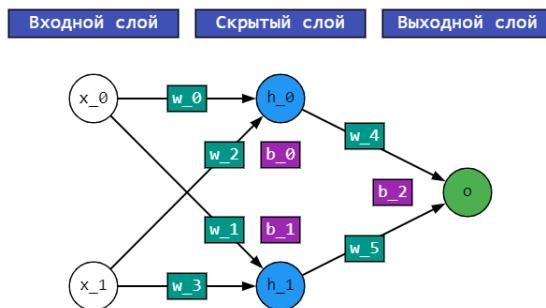


Рис. 1: Архитектура многослойного перцептрана.

К качестве функции активации возьмем сигмоиду $f(x) = 1/(1 + e^{-x})$. Она «упаковывает» интервал от $-\infty$ до $+\infty$ и выдает результаты в интервале $(0, 1)$.

В данной работе постараемся обучить перцептрон до первого вида обучения, когда сеть развивается точно так же, как исходная система, начиная с любой начальной конфигурации, из статьи [5]. Начальные веса будут генерироваться от -10 до 10, а пороги - от -4 до 1.

Алгоритм обратного распространения ошибки

В статье [5] предлагается обучить нейронную сеть с помощью дельта-правила, однако рассмотрим его улучшение – **алгоритм обратного распространения ошибки**, который является алгоритмом машинного обучения на основе градиентного спуска по средней квадратичной ошибке выходного слоя.

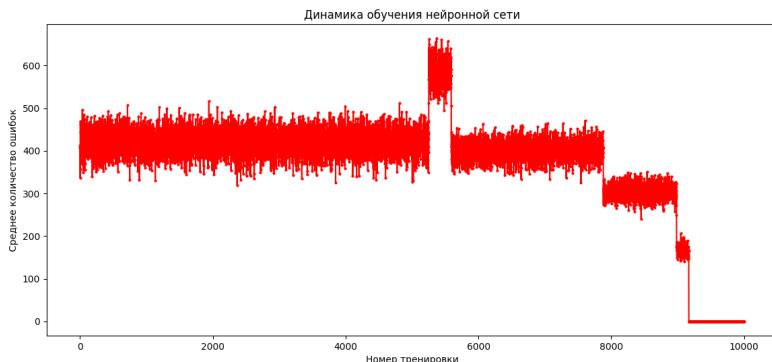


Рис. 2: график удачной попытки обучения на 10000 эпох.

Попытки обучения без лимитов поколений оказались неэффективны, потому что нейросеть долго обучалась правилу появления «живых» клеток, так как в большинстве примеров преобладали «мертвые», которые тянули веса и пороги в сторону обучения правил о них, и не могла «понять», что «живые» клетки могут умирать от перенаселения, потому что примеров с этим явлением было крайне мало.

Для того чтобы решить эти проблемы и ускорить обучение, стоит поставить лимит 2, потому что в нулевом поколении преобладают «живые» клетки, а в первом - «мертвые», что поможет сети параллельно «выучить» правила для них обоих. Также ускорение позволило быстрее выявлять проблемы алгоритма (застревание в локальных минимумах функции ошибки и «паралич сети») и перезапускать обучение.

Коэф.	Значение
w_0	0.11068505585632368
w_1	1.6140490074482765
w_2	1.1231821052847883
w_3	2.1197172145813132
w_4	-11.069308691851203
w_5	11.417776366120652
b_0	-3.9229089223740177
b_1	-4.201344761961087
b_2	-5.64691200600882

Таблица 1: Веса и пороги «обученной» нейронной сети с алгоритмом обратного распространения ошибки

Главная проблема – постоянная скорость обучения, зависящая от градиента ошибки. 1 успешное обучение на 150 попыток.

Удачная попытка обучения представлена на рисунке 2. А её веса и пороги приведены в таблице 1. Оптимальная скорость обучения варьируется от 0.2 до 0.5.

Модифицированный алгоритм устойчивого обратного распространения с уменьшением параметров

Справиться с недостатками прошлого алгоритма может **модифицированный алгоритм устойчивого обратного распространения с уменьшением параметров** [6], который использует только знаки частных производных для подстройки параметрических коэффициентов $\Delta_{i,j}$ и действует независимо для каждого параметра.

Если на текущем шаге частная производная по соответствующему весу или порогу поменяла свой знак, то это говорит о том, что последнее изменение было большим, и алгоритм проскочил локальный минимум. Однако выполнить «откат» будет контрпродуктивно, если общая ошибка уменьшилась. Данный алгоритм объединяет “индивидуальную” информацию о поверхности ошибки с ошибкой сети, чтобы решить для каждого параметра в отдельности, следует ли отменять шаг.

Алгоритм показал себя лучше предыдущего в скорости обучения. Проблемы застrevаний в локальных минимумах функции ошибки также возник-

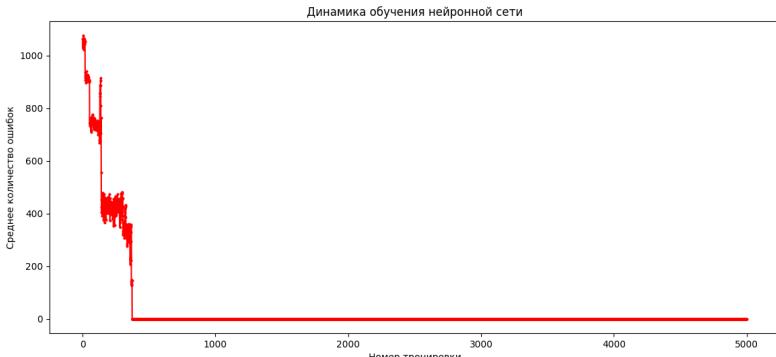


Рис. 3: график удачной попытки обучения на 5000 эпох.

ли, однако алгоритм «выбирался» из малых минимумов. И «параличей сети» не возникло: алгоритм возвращался из области насыщения с помощью параметрических коэффициентов. Однако процент успешных обучений вырос лишь немного: 3 успешных на 150 попыток.

Одна из удачных попыток обучения представлена на рисунке 3. А её веса и пороги приведены в таблице 2. В ходе работы были выявлены оптимальные параметры $\theta^+ = 1.2$, $\theta^- = 0.5$, $\Delta_{min} = 0.00001$, $\Delta_{max} = 0.01$, а начальный параметрический коэффициент для каждого веса и порога 0.0005.

Заключение

В процессе работы над статьей был реализован многослойный перцептрон с двумя алгоритмами его обучения. Был предложен способ сократить время обучения, выставив лимит поколений на каждом этапе до 2. По результатам может быть сделан вывод, что несмотря на то, что игра «Жизнь» – самый примитивный клеточный автомат, но тем не менее обладает непростой производной среднеквадратичной ошибки, что приводит к трудностям в процессе обучения нейронных сетей. Алгоритмы полагаются на удачные случайные начальные веса и пороги, которые быстро сходятся к решению [7]. Из этого следует, что среднеквадратичная ошибка имеет большое количество минимумов разной «глубины».

С учетом этой сложности, предложим два варианта решения проблемы. Первый - бесконечное запускание модифицированного алгоритма устойчивого обратного распространения с уменьшением параметров со случайными весами и порогами и лимитом поколений, пока не будет найден глобальный

Коэф.	Значение
w_0	1.7747016838681917
w_1	0.18769040773208562
w_2	2.391405692710275
w_3	1.2853763296893144
w_4	42.575969840202355
w_5	-57.147982604809194
b_0	-5.276672938117282
b_1	-4.997180607315179
b_2	-17.444888764084585

Таблица 2: Веса и пороги «обученной» нейронной сети с модифицированным алгоритмом устойчивого обратного распространения с уменьшением параметров

минимум. Второй - создание идеального в теории алгоритма для игры, который сможет сканировать все локальные минимумы с возможностью «выпрыгнуть» на следующий при не нахождении там глобального минимума.

Кроме того, возможно улучшение архитектуры нейронной сети до сетей Колмогорова-Арнольда [8], которая может обеспечить более эффективное обучение. Помимо этого допустим переход на Q-обучение, основанный на выборе следующего поколения с помощью так называемой Q-таблицы, что является аналогом обучения с учителем для выбранной игры.

Список литературы

- [1] Wolfram S. «Statistical Mechanics of Cellular Automata». Reviews of Modern Physics, 2013. <https://content.wolfram.com/publications/2020/08/statistical-mechanics-cellular-automata.pdf>
- [2] Сериков Д., Очоа Бикэ А.. «Теория клеточных автоматов как метод описания процесса кристаллизации урана». Electronic archive of Tomsk Polytechnic University. https://earchive.tpu.ru/bitstream/11683/17630/1/conference_tpu-2015-C49-097.pdf
- [3] Gardner, M. «The fantastic combinations of John Conway's new solitaire game 'life'». Scientific American 223, 1970. <https://web.stanford.edu/class/sts145/Library/life.pdf>

- [4] Heaton J.. «Introduction to Neural Networks With Java». Heaton Research, 2008. <https://typeset.io/pdf/introduction-to-neural-networks-with-java-3yzltarx3p.pdf>
- [5] Wulff N. H., Hertz J. A.. «Learning cellular automaton dynamics with neural networks». NIPS'92: Proceedings of the 5th International Conference on Neural Information Processing Systems, 1992. <https://proceedings.neurips.cc/paper/1992/file/d6c651ddcd97183b2e40bc464231c962-Paper.pdf>
- [6] Igel C., Husken M..«Empirical evaluation of the improved Rprop learning algorithms»Institut für Neuroinformatik, Ruhr-Universität Bochum, 2001. <https://sci2s.ugr.es/keel/pdf/algorithm/articulo/2003-Neuro-Igel-IRprop+.pdf>
- [7] Springer J., Kenyon. G. «It's Hard for Neural Networks To Learn the Game of Life». International Joint Conference on Neural Networks (IJCNN), 2021. <https://openreview.net/pdf?id=uKZsVyFKbaj>
- [8] Liu Z., Wang Y., Vaidya S., Ruehle F., Halverson J., Soljačić M., Hou T., Tegmark M.. «KAN: Kolmogorov-Arnold Networks». arXiv:2404.19756, 2024. <https://arxiv.org/pdf/2404.19756>

Пространственный анализ данных для исследования потенциала территории

Плотникова М.А., СПбГУ, Санкт-Петербург st087861@student.spbu.ru

Аннотация

В данной работе рассматривается задача выбора наилучшего местоположения для открытия новой точки оператора сотовой связи "Билайн" на территории города Москвы. Потенциальные участки получены путем разбиения географической карты на гексагоны и последующего анализа их привлекательности на основе метода Вороного. В исследовании используются данные из открытых источников, таких как Яндекс.Карты и Реформа ЖКХ. Предложены 10 наилучших мест для открытия магазина, результаты представлены на интерактивной карте.

Введение

При выборе открытия наилучшей точки, ритейлером должны учитывать множество факторов. Существует несколько подходов [1], [2] к решению данной задачи. В данной работе стояла цель использовать только данные из открытых источников, поэтому был выбран подход на основе геопространственного анализа. Этот подход находит широкое применение в различных отраслях (например, проект по установке солнечных батарей [3], построению зданий [4] или заправочных станций [5] на местности), поскольку он универсален и интерпретируем, что делает его подходящим инструментом для принятия обоснованных решений для бизнеса.

Алгоритм, представленный в этой работе, будет сосредоточен на расчете универсальных метрик с использованием географических и демографических данных.

Конкретно использовались данные о географическом местоположении текущих магазинов и о плотности населения, информация о которых была выгружена из "Яндекс.Карт" и данных проекта "Реформа ЖКХ" соответственно. На основе выгруженных данных была построена метрика, оценивающая доступность магазинов для населения, и предложены наилучшие географические зоны для улучшения покрытия.

В работе используются такие библиотеки python, как geopandas, numpy, osmnx, folium и shapely. Реализованы интерактивные карты для удобного взаимодействия с данными.

Выгрузка данных

Для анализа использовались следующие источники данных:

Open Street Map [7] представляет собой некоммерческий веб-картографический проект по созданию силами сообщества участников – пользователей интернета подробной свободной и бесплатной географической карты мира.

С помощью библиотеки OSMnx [8] были выгружены административные границы города Москвы в виде объекта Polygon библиотеки shapely.

Яндекс.Карты [9] - поисково-информационная картографическая служба Яндекса. Сервис предоставляет фиксированное число бесплатных поисковых запросов через API к основной системе поиска.

Использовался тип запроса "Поиск по организациям" для выгрузки данных о магазинах операторов сотовой связи. Выгруженные данные:

- название магазина;
- адрес;
- координаты точки на карте мира;

Реформа ЖКХ [10] - автоматизированная информационная система, созданная по заказу Фонда содействия реформированию ЖКХ, ориентированная на мониторинг региональных программ переселения граждан из аварийного жилья, программ капитального ремонта многоквартирных домов. Выгруженные данные:

- площадь жилых помещений, площадь нежилых помещений в м^2 ;
- адрес здания;
- координаты здания.

Расчет проживающих в здании производился по формуле:

$$\left(\frac{\text{area} \times 0.35}{33} + \frac{\text{area} \times 0.65}{18} \right) \quad (1)$$

Где area - это площадь жилого дома. Формула была составлена, исходя из норм для проживающих в однокомнатных квартирах с большим числом комнат соответственно - 33м^2 для проживающих в однокомнатной квартире, и 18м^2 - для семей.

Разбиение территории на полигоны

После выгрузки границ Москвы, было построено покрытие гексагонами библиотеки h3. 1

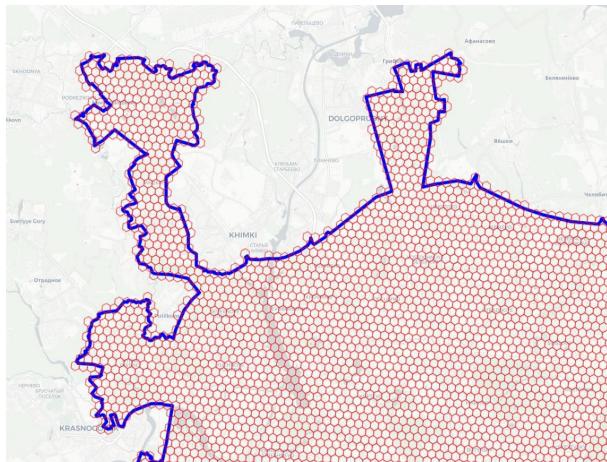


Рис. 1: Построение покрытия города Москвы гексагонами

Гексагон, обладающий наиболее подходящими характеристиками, в будущем будет выбран в качестве решения поставленной задачи.

Используемые python библиотеки: numpy [11], pandas [12], geopandas [13], h3[14].

Целью при обработке географических данных являлась оценка качества доступности торговых точек для населения.

Был применен метод Вороного [6] для разбиения карты в зависимости от удаленности до точки. Полученное разбиение было пересечено с полигонами. Для каждого полигона полученного разбиения была посчитана дистанция до ближайшего магазина, рассчитанная, как максимальное расстояние из полигона до точки (метрика Хаусдорфа). Используемые python библиотеки: geovoronoi [15], shapely [16], folium [17].

Оценка доступности торговых точек для населения

Полученные полигоны были пересечены с координатами населенных домов. Итоговая метрика ρ была подсчитана по формуле

$$\rho = \frac{p}{d}$$

где p - это плотность населения на участке, d - максимальное расстояние до ближайшего магазина.

В случае, когда гексагон пересекался с разбиением Вороного и имел несколько наиболее близких к нему точек оператора "Билайн", метрика подсчитывалась отдельно для каждого участка, и затем усреднялась для всего гексагона. Результаты были сформированы в виде файла csv и выведены на интерактивную карту 2.

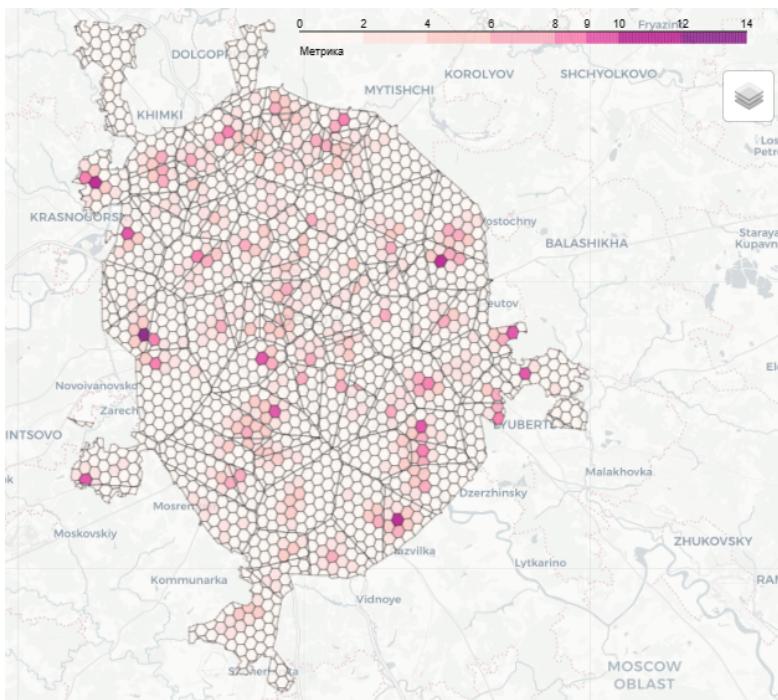


Рис. 2: Гексагоны окрашены в зависимости от значения построенной метрики.

Улучшение покрытия

В качестве наиболее подходящих мест для открытия новых точек были выбраны 10 значений с наибольшим значением метрики.

В центрах данных гексагонов были созданы новые точки, которые в дальнейшем участвовали в повторном разбиении 3.

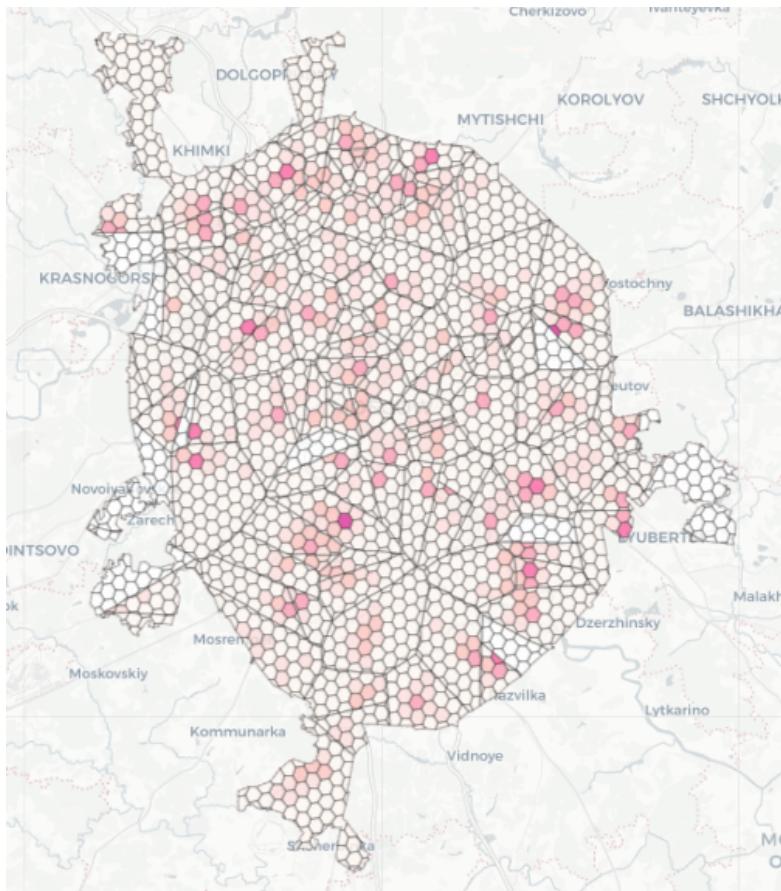


Рис. 3: Карта нового разбиения, построенная после добавления 10 дополнительных точек. Гексагоны окрашены в зависимости от значения построенной метрики.

На карте наглядно продемонстрировано улучшение качества покрытия после добавления 10 новых точек.

Приложение

Код, разработанный в рамках данного исследования, а также интерактивные карты доступны по ссылке:

https://github.com/Ima0203/GEO_searching

Заключение

В ходе работы были исследованы существующие подходы в решении подобных задач, реализована загрузка и обработка данных из открытых источников с использованием API технологии и библиотек языка python. Был проведен отбор наиболее качественных из них и подходящих для решения задачи.

На основе алгоритма Вороного и разбиения территории на гексагоны, была разработана метрика доступности магазинов для населения. На основе ее показателей, была реализована интерактивная карта города Москвы с указанием наиболее подходящих местоположений для открытия новой точки. Решение было представлено с помощью интерактивных карт folium.

Список литературы

- [1] Grin, Petr. *Utilization of machine learning algorithms to support retail chain store location decisions*. University of Northern Iowa, 2020.
- [2] Ting, Choo-Yee, Mang Yu Jie. *Location Profiling for Retail-Site Recommendation Using Machine Learning Approach*, 2020.
- [3] Al Garni, H. Z., Awasthi, A. *Solar PV Power Plant Site Selection Using a GIS-AHP Based Approach with Application in Saudi Arabia*. Applied Energy, 2017.
- [4] Kumar, S., Bansal, V. *A GIS-based Methodology for Safe Site Selection of a Building in a Hilly Region*. Frontiers of Architectural Research, 2016.
- [5] Messaoudi, D., Settou, N., Negrou, B., Rahmouni, S., Settou, B., Mayou, I. *Site Selection Methodology for the Wind-powered Hydrogen Refueling Station Based on AHP-GIS in Adrar, Algeria*. Energy Procedia, 2019.
- [6] *Spatial Tessellations: Concepts and Applications of Voronoi Diagrams*. John Wiley & Sons, 2009.
- [7] «Open Street Map». URL: <https://openstreetmap.org/>
- [8] OSMnx. URL: <https://github.com/gboeing/osmnx>
- [9] «Yandex API». URL: <https://yandex.ru/maps-api/>
- [10] «Открытые данные ЖКХ». URL: <https://dom.mingkh.ru/>
- [11] NumPy. URL: <https://numpy.org/>
- [12] Pandas. URL: <https://pandas.pydata.org/>
- [13] GeoPandas. URL: <https://geopandas.org/>
- [14] H3. URL: <https://github.com/uber/h3-py>
- [15] Geovoronoi. URL: <https://github.com/WZBSocialScienceCenter/geovoronoi>
- [16] Shapely. URL: <https://github.com/Toblerity/Shapely>
- [17] Folium. URL: <https://github.com/python-visualization/folium>

Разработка отказоустойчивого конвейера для доставки данных в режиме квазиреального времени

Сафин А.Ф., СПбГУ, Санкт-Петербург st090196@student.spbu.ru,
Благов М.В., СПбГУ, Санкт-Петербург mikhail.blagov@gmail.com

Аннотация

В статье рассматривается разработка и тестирование отказоустойчивого потокового конвейера данных. В работе описан процесс проектирования и проверки надёжности системы. Особое внимание уделяется рассмотрению возможных сбоев и методам обеспечения отказоустойчивости, таким как использование отказоустойчивых компонентов и автоматическое восстановление после сбоев.

Введение

При решении задачи анализа данных возникает потребность в переносе данных с продуктивной базы данных в аналитическое хранилище, которое было создано специально для того, чтобы потреблять и структурировать данные для последующего анализа [1].

Аналитическое хранилище облегчает анализ данных, не создавая нагрузки на источники, которые обеспечивают повседневную работу. Обычно, данные в аналитическом хранилище обновляются периодическим образом, например, еженедельно или ежедневно. Загрузка данных в аналитическое хранилище, как правило, запланирована на непиковые часы, когда как источники, так и хранилище испытывают низкую нагрузку, например, в ночное время. Таким образом, оно хранит данные по состоянию на прошлую неделю или на вчерашний день, в то время как текущие данные доступны только в источниках [2].

Однако бизнес-пользователи часто нуждаются в актуальных данных для поддержки своевременного принятия решений. Данная задача решается при помощи потоковых систем обработки данных, которые позволяют в режиме квазиреального времени переносить данные из источника в аналитическое хранилище [2]. На практике поток данных часто не ограничен, информация приходит постепенно, с течением времени [3]. Важно обработать каждое сообщение в отдельности таким образом, чтобы оно хотя бы один раз [4] дошло из источника в хранилище.

Также необходимо, чтобы доставка данных происходила с минимальными задержками и максимальной надежностью. Данная работа посвящена тому, чтобы разработать отказоустойчивый потоковый конвейер обработки данных, используя популярные инструменты, и показать его устойчивость к сбоям.

Проектирование конвейера данных

Целью работы является построение и реализация принципов отказоустойчивости конвейера для доставки данных в режиме квазиреального времени, которые позволяют потоковому приложению продолжать функционировать надежно и эффективно даже при возникновении сбоев. Это включает в себя проектирование системы таким образом, чтобы она могла обнаруживать и восстанавливаться от сбоев.

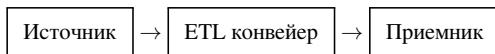


Рис. 1: Верхнеуровневая схема конвейера данных

Рассмотрим конвейер данных, изображенный на рис. 1. Такой конвейер данных является распределенной системой. Рассмотрим возможные типы неисправностей в распределенных системах.

Типы неисправностей в распределенных системах

Распределенные системы характеризуются высокой сложностью из-за большого количества компонентов и использованием различных аппаратных и программных платформ [5]. Эта сложность делает их склонными ко многим проблемам. Выделяют три типа проблем, которые могут возникнуть в распределенной системе:

- Сбой – это неожиданное или ненормальное поведение компонента системы, которое может привести к ошибке или сбою [5].
- Ошибка – ошибочное состояние системы, возникающее в результате сбоев [5].
- Отказ определяет событие, при котором система не может предоставить услугу или выполнить намеченную цель. Это видимый результат ошибки [5].

Сбой в одном узле может распространяться по всей системе. Поэтому важно обеспечить, чтобы она продолжала работать даже при наличии неисправностей. Это достигается путем предварительного анализа типов и учетом этих неисправностей при разработке системы [5].

Неисправности классифицируются по частоте их появления на временные, периодические и постоянные неисправности. Временные неисправности возникают один раз и исчезают, а периодические неисправности появляются и исчезают неоднократно. Что касается постоянных неисправностей, то они появляются и остаются до тех пор, пока не будут устранены [5].

Определение: Отказоустойчивость – это способность конвейера данных как распределенной системы функционировать должным образом даже при наличии временных и периодических сбоев [5].

Для конвейера, изображенного на рис. 1, реализуем принципы отказоустойчивости. Для него характерны следующие типы отказов:

- сбой источника данных;
- сбой сети между источником и ETL;
- сбой отдельных компонентов, виртуальных машин и контейнеров, на которых запущен ETL;
- сбой сети между ETL и приемником;
- сбой приёмника.

Будем рассматривать временные и периодические сбои, при которых функционирование компонента восстанавливается спустя некоторое время после первичного сбоя.

Архитектура конвейера обработки данных

На рис. 2 представлена схема конвейера. В ней можно выделить следующие элементы:

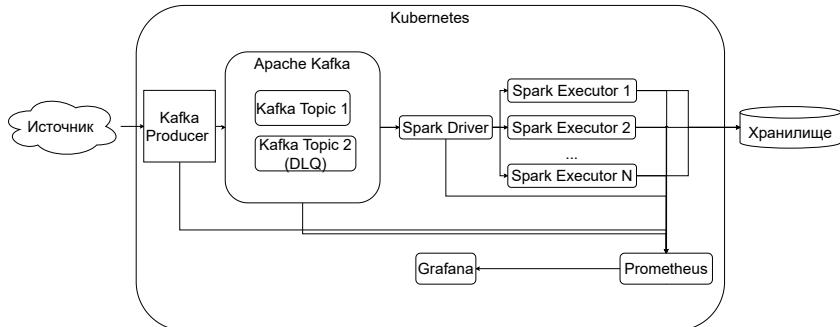


Рис. 2: Архитектура

- Источник – система, из которой конвейер читает данные;
- Набор блоков для обработки данных, развёрнутых в Kubernetes [6]:
 - Kafka Producer читает данные из источника и отправляет сообщения в Kafka.
 - Блок Kafka [7] отвечает за хранение сообщений, которые предстоит обработать: сообщения из Kafka Topic [8] 1 отправляются в Spark приложение, Kafka Topic 2 (DLQ) содержит сообщения, которые были некорректно обработаны и требуют повторной обработки для обеспечения “хотя бы один раз” семантики доставки данных.
 - Spark Driver [9] отвечает за три вещи: сохранение информации о приложении Spark; реагирование на программу пользователя или вводимые данные; и анализ, распределение и планирование работы между исполнителями.
 - Spark executors [9] несут ответственность за фактическое выполнение работы, которую им поручает Spark Driver. Это означает, что каждый Spark executor отвечает только за две вещи: выполнение кода, назначенного ему Spark Driver’ом, и отправку отчета о состоянии вычислений на этом исполнителе обратно на узел Spark Driver’а.
 - Блок мониторинга отвечает за наблюдаемость для конвейера, а также служит для оповещения о сбоях. Данный блок реализован при помощи инструментов Prometheus [10], Grafana [11].
 - Система-приемник, в которую конвейер должен доставить данные.

Методика тестирования

Для тестирования отказоустойчивости применим подход, известный как Хаос-тестирование [12, 13]. Приложение будет работать какое-то время при "идеальных" условиях. Далее, искусственно будем вносить в систему сбои и отслеживать, как контейнер реагирует на это.

- При сбое источника данных производится вывод предупреждения и повторная попытка соединения. После восстановления источника конвейер переподключается к нему автоматически.
- При отсутствии связи с источником Kafka Broker [7] продолжает функционировать, но не обрабатывает данные, поэтому необходимо просигнализировать о падении входного потока сообщений. После восстановления соединения конвейер продолжает работу.
- При сбое Kubernetes необходим перезапуск кластера вручную. При этом связь между подами [6] может потеряться. Использование StatefulSets [6] вместо стандартных Deployments [6] исключает обозначенную проблему.
- При сбое Kafka Producer, под автоматически перезапускается. При этом возможна потеря сообщений. Использование механизма повторной отправки (retry mechanisms) [7] исключает обозначенную проблему.
- При сбое Spark Driver, под автоматически перезапускается, при этом нужно не потерять данные. Для этого необходимо дать Spark инструкцию фиксировать offsets [8] только после того, как произведена операция записи в хранилище. Необработанные же сообщения (которые не были записаны в хранилище) отправляются в резервный топик Kafka для повторной обработки.
- При сбое одного из Spark Executor он автоматически перезапускается, необработанные сообщения отправляются в резервный топик Kafka для повторной обработки. При этом оставшийся Executor автоматически продолжает обработку сообщений.
- При отсутствии связи с приемником, ETL конвейер продолжает функционировать, но сообщения отправляются в DLQ топик как необработанные. Поэтому необходимо просигнализировать о падении выходного потока сообщений. После восстановления соединения конвейер продолжает работу.

- При сбое приемника производится вывод предупреждения и повторная попытка соединения. После восстановления приемника конвейер переподключается к нему автоматически.

Заключение

В работе сформулированы принципы отказоустойчивости конвейера для доставки данных в режиме квазиреального времени. Рассмотрены основные типы отказов для таких конвейеров, реализовано потоковое приложение, проведено тестирование путем внесения временных сбоев в систему и показана отказоустойчивость разработанного конвейера в рассмотренных случаях.

Список литературы

- [1] Что такое потоковая передача данных? // <https://aws.amazon.com/ru/what-is/streaming-data/>
- [2] Jörg, T., Dessloch, S. Near Real-Time Data Warehousing Using State-of-the-Art ETL Tools // Lecture Notes in Business Information Processing. – Springer, 2009. – С. 100–117.
- [3] Клеппман, М. Высоконагруженные приложения. Программирование, масштабирование, поддержка. – Санкт-Петербург: Питер, 2018. – 640 с.
- [4] Bhimani, P., Panchal, G. Message Delivery Guarantee and Status Update of Clients Based on IoT-AMQP // Lecture Notes in Networks and Systems – 2017. – С. 15-22.
- [5] Rhim, H. Fault and Failure in Distributed Systems // <https://www.baeldung.com/cs/distributed-systems-fault-failure>.
- [6] Hightower, K., Burns, B., Beda, J. Kubernetes: Up and Running. 2nd edition. – Sebastopol, CA: O'Reilly Media, Inc., 2019. – 277 с.
- [7] Kreps, J. Putting Apache Kafka to Use: A Practical Guide to Building a Stream Data Platform (Part 2). February 25, 2015. <https://www.confluent.io/blog/stream-data-platform-2/>
- [8] Скотт, Д., Гамов, В., Клейн, Д. Kafka в действии. 1-е издание. – Москва: ДМК Пресс, 2022. – 310 с.

- [9] Optimizing Spark performance on Kubernetes // <https://aws.amazon.com/ru/blogs/containers/optimizing-spark-performance-on-kubernetes/>
- [10] What is Prometheus? // <https://prometheus.io/docs/introduction/overview/>
- [11] What is Grafana? // <https://www.redhat.com/en/topics/data-services/what-is-grafana>
- [12] Principles of Chaos Engineering // <https://principlesofchaos.org/>
- [13] Lewis, J., Wang, C. Chaos Engineering: New Approaches To Security // Rain Capital, 2019.
- [14] White, T. Hadoop: The Definitive Guide STORAGE AND ANALYSIS AT INTERNET SCALE. 4th edition. – Sebastopol, CA: O'Reilly Media, Inc., 2015. – 728 c.

Распараллеливание в OpenMP и сплайновые аппроксимации



Бурова Ирина Герасимовна

д.ф.-м.н., профессор, профессор кафедры вычислительной математики

Stochastic properties of a random Young diagram

Yang Yu., SPBU, Saint-Petersburg st080797@student.spbu.ru,
 Yakubovich Y., SPBU, Saint-Petersburg y.yakubovich@spbu.ru

Abstract

This paper investigates the stochastic properties and dynamic evolution of random Young diagrams using advanced computer simulations, focusing on the application of the Corner_w model with $w = 1$. We explore how variations in initial parameters affect the growth and limit shapes of these diagrams. Our simulation methodology constructs Young diagrams incrementally, following probabilistic rules derived from theoretical models such as Rost's limit shape theory.

We implement a novel color-coding technique to visually represent the frequency of individual boxes within the diagrams, using a gradient from dark green to white to indicate varying frequencies. This method clarifies the statistical distribution of elements within the diagrams and enhances our understanding of their probabilistic behaviors over multiple iterations. Our findings show that as the number of iterations increases, the emergent Young diagrams exhibit a stable limit shape consistent with theoretical predictions.

Additionally, our simulations reveal the aggregate behavior of multiple Young diagrams, providing new insights into their collective dynamics and structural properties when observed as an ensemble. The higher frequency of box appearances is indicated by lighter shades, moving towards white for boxes appearing in every iteration, thus offering an intuitive and interactive way to analyze these complex structures.

In the theoretical exploration of jeu de taquin, we developed new combinatorial insights by examining the interaction dynamics of dual paths within a Young tableau. Our findings reveal specific forbidden movement patterns and interaction scenarios between these paths, enhancing the understanding of their complex behavior. Key results include proofs of several lemmas detailing these forbidden configurations and their implications for the transformation dynamics of jeu de taquin paths.

Additionally, we propose a conjecture on the stability of endpoint distributions under the jeu de taquin transformation, supported by empirical data from the simulations. This conjecture suggests

predictable patterns in the development of jeu de taquin paths, potentially contributing significantly to the theoretical framework of combinatorial dynamics in Young tableaux.

These contributions extend the current understanding of Young diagrams and jeu de taquin dynamics, opening new avenues for research in probabilistic and combinatorial mathematics.

References

- [1] Dan Romik and Piotr Sniady, JEU DE TAQUIN DYNAMICS ON INFINITE YOUNG TABLEAUX AND SECOND CLASS PARTICLES, *The Annals of Probability*, 2015, Volume 43, Pages 682–737, Number 2.
- [2] Eriksson, K. and Sjöstrand, J., Limiting shapes of birth-and-death processes on Young diagrams, *Advances in Applied Mathematics*, Volume 48, Pages 575–602, Year 2012.
- [3] A. M. Vershik, Statistical mechanics of combinatorial partitions, and their limit shapes, *Funct. Anal. Appl.*, 1996, Volume 30, Pages 90–105.
- [4] H. Simon, On a class of skew distribution functions, *Biometrika*, 1955, Volume 42, Pages 425–440.
- [5] H. Rost, Non-equilibrium behaviour of a many particle process: Density profile and local equilibria, *Probab. Theory Related Fields*, 1981, Volume 58, Pages 41–53.
- [6] W.J. Ewens, *Mathematical Population Genetics*, Springer, 2004, Edition second.
- [7] Gideon Amir and Omer Angel and Benedek Valkó, The TASEP Speed Process, *The Annals of Probability*, 2011, Volume 39, Number 4, Pages 1205–1242, DOI: 10.1214/10-AOP561.
- [8] Mountford, T. and Guiol, H., The motion of a second class particle for the TASEP starting from a decreasing shock profile, *Ann. Appl. Probab.*, 2005, Volume 15, Pages 1227–1259.

Архитектура динамической децентрализованной большой языковой модели с использование технологии консенсусов

Козгунов Н., СПбГУ, Санкт-Петербург st090866@student.spbu.ru,
Халаша М., СПбГУ, Санкт-Петербург st082966@student.spbu.ru¹

Аннотация

В данной исследовательской работе представлена новая концепция большой языковой модели с эффективным использованием вычислительных мощностей посредством использования механизмов управления на основе технологии консенсусов и блокчейна, предоставляя стимулы пользователям транзакциями на основе монет, тем самым позволяя возможность масштабируемости. Одновременно, предлагая более защищенный подход для обработки данных и дообучения модели, устранив проблему единой точки сбоя с помощью децентрализованного федеративного обучения модели, представленная система решает проблемы существующих больших языковых моделей.

Введение

Ранние 2010-е годы ознаменовались значительным прогрессом в области обработки естественного языка, начиная с появления Word2Vec, который заложил основу для последующих инноваций в этой области. Эти ранние достижения привели к разработке ELMo и были ключевыми в создании архитектуры Transformer, вдохновив на развитие больших языковых моделей (LLM), таких как BERT и серия GPT. Эти последовательные инновации совместно повысили понимание синтаксиса и семантики, устанавливая новые стандарты для понимания языка. Одновременно с этим эволюция блокчейна, начавшаяся с Bitcoin и далее развившаяся с Ethereum, преобразила цифровую валюту и интернет в модель, ориентированную на пользователя, Web 3.0. Это развитие, как подробно описывается, расширило его применение до децентрализованных приложений (DApps) и невзаимозаменяемых токенов (NFT). В данной работе рассматривается слияние LLM и технологии блокчейн, предлагая новую архитектуру, которая объединяет лингвистические возможности, выделенные в работах, с децентрализованной природой блокчейна. Это предложение направлено на решение проблем, связанных с конфиденциальностью, предвзятостью и централизацией. Кроме того,

¹Авторы внесли равный вклад в данную работу.

оно выступает за использование сети блокчайна для сокращения развивающихся затрат, связанных с LLM, стремясь к масштабируемой и устойчивой модели, которая демократизирует доступ к технологиям ИИ и способствует инклюзивному технологическому будущему.

Обзор литературы

Централизованные системы федеративного обучения демонстрируют значительные преимущества, в том числе обеспечение конфиденциальности данных и оптимизацию управлеченческих процессов, однако они подвержены определенным недостаткам, включая коммуникационные издержки и риски единой точки сбоя, обусловленные зависимостью от централизованных серверов [1]. В качестве альтернативы представляется децентрализованное федеративное обучение, которое внедряет механизмы, основанные на гомоморфном шифровании, для обеспечения безопасного и эффективного обучения через множество источников без необходимости расшифровки, тем самым улучшая устойчивость и производительность системы [2], а также предлагая структуру, основанную на DAG, для обеспечения надежности децентрализованных операций, стимулируя более широкое участие и обеспечивая справедливые стимулы участникам [3]. Потенциал интеграции федеративного обучения и децентрализованных методов с большими языковыми моделями (LLM) значителен. Исследование FusionAI демонстрирует, что потребительские графические процессоры могут эффективно использоваться для обучения и развертывания LLM, предлагая экономически выгодную альтернативу [4]. Федеративные LLM адресуют проблемы дефицита данных и приватности в искусственном интеллекте, применяя колаборативное обучение, включая процессы предварительного обучения и дообучения, в рамках централизованной модели FL [5]. Предлагаемая архитектура способствует созданию дополнительной оптимизации процесса вывода LLM на мобильные устройства, повышая быстродействия системы [6]. Улучшение функциональности LLM на индивидуальных узлах, особенно в распределенных конфигурациях, способствует повышению общей эффективности системы. Разработка FlexGen обеспечивает высокопроизводительный движок для генеративного вывода в условиях ограниченных ресурсов [7], в то время как адаптация LLM для работы на устройствах с ограниченной оперативной памятью, например, через использование флэш-памяти для хранения данных, значительно ускоряет процесс вывода [8]. Трансформации в блокчейн-технологиях являются ключевыми, с переходом от механизма доказательства выполнения работы (Proof-of-Work, PoW) к механизму доказательства доли (Proof-

of-Stake, PoS), что вносит фундаментальные изменения в механизмы достижения консенсуса через владение долями, имея критическое значение для будущей безопасности и производительности сетей [9]. Механизм доказательства вовлечения (Proof-of-Engagement, PoE) предоставляет гибкую систему консенсуса, акцентируя внимание на активных вкладах узлов, тем самым продвигая принципы равенства [10], в то время как делегированное доказательство репутации (Delegated Proof-of-Reputation) объединяет PoS с системой репутации для повышения масштабируемости и безопасности [11], что отражает динамичное развитие технологии блокчейна и протоколов консенсуса. Таким образом, интеграция методов машинного обучения с блокчейн-технологией представляет собой перспективное направление развития. Современный подход к децентрализации алгоритмов машинного обучения и данных через использование блокчейна направлен на усиление безопасности, приватности и коллаборативности. В контексте этих разработок сочетание больших языковых моделей и блокчейн-технологий открывает новые возможности, совмещая контекстуальное понимание и творческие способности LLM с безопасностью и прозрачностью блокчейна, предлагая эффективное синергетическое взаимодействие [12]. Эта интеграция рассматривается через оценку ее преимуществ и вызовов, предлагая архитектурные решения для инкорпорации LLM в децентрализованные приложения, трансформируя неоднозначные намерения в четкие и выполнимые директивы.

Архитектура динамической децентрализованный

Обзор

Веса в предлагаемой архитектуре первоначально устанавливаются с помощью методов обучения с передачей данных, которые затем распространяются по сети в одноранговом режиме. За этим шагом следует локальное обновление модели с использованием подходов федеративного обучения для уточнения модели на децентрализованных узлах. Затем эти локальные модели объединяются в новую, расширенную версию LLM, которая снова распространяется по сети. В основе всего этого процесса лежит блокчейн, облегчающий обновление и контроль версий LLM и включающий криптовалютную схему поощрения для мотивации участия узлов.

Компоненты

Предлагаемая архитектура модели состоит из следующих элементов:

- Nodes: Конечные устройства, которые участвуют в сети, предоставляя вычислительные ресурсы и данные.
- Learning Unit: Обновленные веса модели, полученные узлами в результате локального обучения на своих наборах данных.
- Data Unit: Фрагмент набора данных, который узлы решают передать сети.
- Transaction Unit: Обмен монетами между узлами в сети.
- Memory Pool (Mempool): Место для хранения learning units, data units и transactions перед их компиляцией в новые блоки.
- Transaction Block: Блок, состоящий из нескольких транзакций, выбранных из пула памяти.
- Data Block: Блок, включающий транзакцию coinbase для получения вознаграждения и компиляцию блоков данных из Mempool, формирующих тестовый набор данных.
- Version Block: Блок, содержащий транзакцию coinbase за вознаграждение и агрегированные веса, формирующие глобальную модель, полученную из комбинации единиц обучения из Mempool.
- LinguaChain: Блокчейн, состоящий из transaction blocks, data blocks и version blocks, последовательно связанных между собой с помощью хешей предыдущих блоков.

Упрощенная схема предлагаемой архитектуры LinguaChain представлена на (см. Рис. 1).

Описание работы

Архитектура начинается с установки начальных весов на выбранном узле путем трансферного обучения, используя веса из предварительно обученной модели, а не случайным выбором. Затем эти веса распространяются по одноранговой сети (P2P), позволяя узлам индивидуально обучать модель на своих данных. После обучения узлы обновляют веса модели и инкапсулируют эти обновления в блок обучения. Этот блок защищается с помощью шифрования с закрытым ключом, создавая уникальную цифровую подпись, и впоследствии отправляется в Mempool. Для поддержки последующего комбинирования моделей применяется гомоморфное шифрование. Одновременно

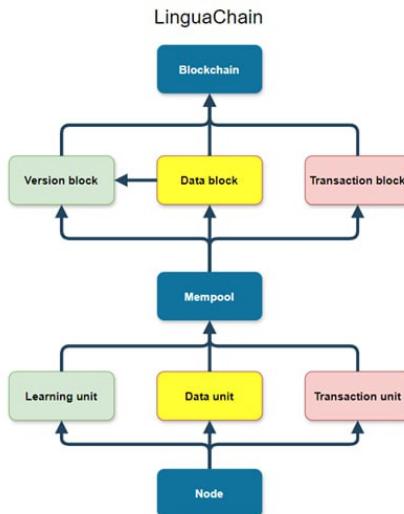


Рис. 1: Архитектура LinguaChain.

узлы совершают криптовалютные транзакции в блокчейне, и эти транзакции также направляются в Мемпools. Более того, некоторые узлы предоставляют тестовые наборы данных, шифруя и подписывая единицы данных в процессе, схожем с процессом обучения, перед тем как отправить их в Мемпools, где гомоморфное шифрование позволяет выполнять последующие операции. После этого начального этапа механизм Proof-of-Stake (PoS) выбирает узлы на основе их криптовалютных холдингов для создания новых блоков, добавляя определенную степень случайности для обеспечения разнообразного выбора создателей блоков. Эти узлы извлекают, проверяют и собирают транзакции из Мемпools в новый блок, который включает транзакцию coinbase [23] в качестве вознаграждения для создателя. Затем этот блок распространяется по сети для проверки целостности. Если большинство узлов подтверждают транзакции, блок интегрируется в блокчейн; в противном случае недействительные транзакции приводят к потере доли создателя блока, что стимулирует тщательную проверку. Система блокчейна устанавливает ограничение на частоту блоков транзакций, требуя добавления блока данных и блока версии перед любыми последующими блоками транзакций. Эта процедура использует комбинацию механизмов PoS, Proof-of-Time (PoT) и Proof-of-Work (PoW). Для генерации блока данных PoS определяет выбор узла на основе ставки, при этом назначается случайный вызов (номер), который обрабаты-

вается с помощью функции верифицируемой задержки (VDF). Узлы выбирают и улучшают блоки данных из пула Metropool для повышения качества набора данных. Создатель набора данных наивысшего качества формирует новый блок данных, который затем аутентифицируется сетью с помощью выходных данных VDF. Создатели и авторы проверенных блоков данных получают вознаграждение через транзакцию coinbase, отражающее их вклад в качество набора данных. Узлы,искажающие качество данных или манипулирующие VDF, рисуют лишиться доли. В соответствии с установленной политикой частоты блоков, блокчейн останавливает включение блоков, не относящихся к транзакциям, до тех пор, пока не будет добавлен блок новой версии. Используя механизм Proof-of-Stake (PoS), узлы выбираются для создания блока версии. Этот процесс отбора, подобный тому, что используется при создании блока данных, направлен на объединение обучаемых блоков для повышения производительности модели. Уникальным аспектом этого этапа является разделение тестового набора данных, полученного из предыдущего блока данных, на различные партии. Затем эти партии распределяются между выбранными узлами. Выдача вызова вместе с этими разнообразными партиями данных побуждает узлы применять усреднение с помощью Federated Averaging (FedAvg) или аналогичными методами для уточнения агрегированной модели на основе полученных данных. После завершения процесса верификации с помощью Verifiable Delay Function (VDF) узлу, достигшему наивысшей производительности по всему набору тестовых данных, поручается создание блока новой версии. После создания этот блок проходит верификацию во всей сети. Успешный процесс проверки не только подтверждает правильность блока версий, но и вызывает вознаграждение как для создателя блока версий, так и для узлов, предоставивших единицы обучения, использованные при его разработке. И наоборот, любой узел, уличенный в злоупотреблениях во время этого процесса, рискует потерять свою долю, что обеспечивает целостность и поощряет честное участие в работе блокчейна. После успешной агрегации и проверки обновленной модели, заключенной в блоке Version блокчейна, узлы получают возможность использовать эту улучшенную модель для решения задач вывода. У них есть возможность либо использовать свои локальные вычислительные ресурсы, что может привести к менее мощным выводам по сравнению с использованием коллективных вычислительных возможностей всей сети, либо получить доступ к более широкой вычислительной мощности и ресурсам сети через механизм обмена монетами с другими узлами. Система поощрений разработана таким образом, чтобы сбалансировать вклад и выгоду для всей сети: предполагается, что средний узел будет зарабатывать на предоставлении единиц данных и единиц обучения примерно столько же, сколько он потратит на доступ к

вычислительным ресурсам для выводов. Этот подход направлен на минимизацию стоимости использования системы для среднего узла. В отличие от этого, крупные узлы, требующие более обширных выводов и высокой точности, могут понести дополнительные расходы на заимствование необходимых вычислительных мощностей из сети. Для обеспечения целостности и конфиденциальности данных в этой архитектуре используются безопасные методы агрегирования, такие как гомоморфное шифрование единиц обучения. Также рассматриваются методы дифференциальной конфиденциальности, которые вводят случайный шум в каждую единицу обучения для дальнейшего усиления защиты конфиденциальности.

Заключение

В данной статье описывается теоретическая архитектура, объединяющая LLM и блокчейн для создания масштабируемого и демократизированного ландшафта ИИ, одновременно решая проблемы конфиденциальности и централизованного контроля. Потенциал такого слияния огромен, оно может повысить эффективность LLM и расширить доступ.

Список литературы

- [1] X. You, X. Liu, X. Lin, J. Cai, S. Chen, "Accuracy Degrading: Toward Participation-Fair Federated Learning" in IEEE Internet of Things Journal, June 2023 DOI: 10.1109/JIOT.2023.3238038.
- [2] A. Bose, L. Bai, "A Fully Decentralized Homomorphic Federated Learning Framework" in IEEE 20th International Conference on Mobile Ad Hoc and Smart Systems (MASS), September 2023, DOI: 10.1109/MASS58611.2023.00029.
- [3] G. Yu, X. Wang, C. Sun, Q. Wang, P. Yu, W. Ni, R. Liu, X. Xu, "IronForge: An Open, Secure, Fair, Decentralized Federated Learning", January 2023, arXiv:2301.04006v1.
- [4] Z. Tang, Y. Wang, X. He, L. Zhang, X. Pan, Q. Wang, R. Zeng, K. Zhao, S. Shi, B. He, X. Chu, "FusionAI: Decentralized Training and Deploying LLMs with Massive Consumer-Level GPUs" in Symposium on Large Language Models (LLM 2023) with IJCAI 2023, Macao, China, August 21, 2023, arXiv:2309.01172.

- [5] C. Chen, X. Feng, J. Zhou, J. Yin, X. Zheng, "Federated Large Language Model: A Position Paper", Zhejiang University, Hangzhou, China, 18 Jul 2023, arXiv:2307.08925v1.
- [6] J. Zhao, Y. Song, S. Liu, I. Harris, S. Jyothi, "LinguaLinked: A Distributed Large Language Model", Dec 2023, arXiv:2312.00388v1.
- [7] Y. Sheng, L. Zheng, B. Yuan, Z. Li, M. Ryabinin, D. Y. Fu, Z. Xie, B. Chen, C. Barrett, J. Gonzalez, P. Liang, C. Re•, I. Stoica, C. Zhang, "FlexGen: High-Throughput Generative Inference of Large Language Models with a Single GPU", Jun 2023, arXiv:2303.06865v2.
- [8] . Alizadeh, I. Mirzadeh, D. Belenko, S. Khatamifard, M. Cho, C. Mundo, M. Rastegari, M. Farajtabar, "LLM in a flash: Efficient Large Language Model Inference with Limited Memory", Jan 2024, arXiv:2312.11514v2.
- [9] C. Nguyen, H. Thai, D. Nyato, N. Nguyen, E. Dutkiewicz, "Proof-of- Stake Consensus Mechanisms for Future Blockchain Networks", IEEE Access, June 2019, DOI: 10.1109/ACCESS.2019.2925010.
- [10] Y. Xu, X. Yang, J. Zhang, J. Zhu, M. Sun, B. Chen, "Proof of Engagement: A Flexible Blockchain Consensus Mechanism" in Wireless Communications and Mobile Computing, Hindawi, August 2021, DOI: 10.1155/2021/6185910.
- [11] T. Do, T. Nguyen, H. Pham, "Delegated Proof of Reputation: a novel Blockchain consensus", December 2019, arXiv:1912.04065v1.
- [12] X. Liu, "Decentralized Machine Learning on a Blockchain: Case Studies" in 2023 IEEE International Conference on Omni-layer Intelligent Systems (COINS), July 2023, DOI: 10.1109/COINS57856.2023.10189230.

Системное программирование и программная инженерия



Терехов Андрей Николаевич

д.ф.-м.н., профессор, заведующий кафедрой системного программирования, президент ООО «Ланит-Терком»



Литвинов Юрий Викторович, к.т.н., старший преподаватель кафедры системного программирования



Луцив Дмитрий Вадимович, к.ф.-м.н., доцент кафедры системного программирования



Сартасов Станислав Юрьевич, к.т.н., старший преподаватель кафедры системного программирования

Построение модели полосы с учетом контекста дороги

Житнухина М.А., СПбГУ, Санкт-Петербург zhitnuhina.maria@gmail.com

Аннотация

В данной работе предлагается модель представления дорожной полосы и алгоритм ее построения по кандидатам на разметку в проекции плоскости дороги «вид сверху».

Введение

В настоящее время активно развивается ADAS. ADAS (англ. Advanced driver-assistance systems) — усовершенствованная система помощи водителю. Эта система руководствуется данными, полученными с датчиков, и через интерфейс сообщает рекомендации пользователю. Для некоторых ее функций, например, автоматической смены полосы движения или удержания в полосе необходима информация о расположении автомобиля в ней.

Возникла необходимость по входным данным, представляющим собой кандидатов на разметку, выделять полосу. Кандидат на разметку — последовательность точек, упорядоченных по удаленности от машины, соответствующая фрагменту дорожной полосы (возможны ложно-положительные срабатывания).

Многие существующие решения используют представление полосы, по которому вычислительно сложно строить траекторию движения, вычислительно сложны в построении самой полосы или недостаточно точно описывают реальность.

Существующие модели полосы

Далее будут рассмотрены существующие модели, используемые для выделения разметки в режиме реального времени.

Множество точек

С множеством точек сложно работать: менее точно вычисляется кривизна, а так же оно занимает много памяти.

Прямые

Апроксимация прямой вычислительно проста, но данная модель не описывает повороты.

Параболы и гиперболы

В статьях [1] нередко можно увидеть использование кривых второго порядка, таких как параболы и гиперболы. Их построение вычислительно не очень сложно и они легко сглаживаются, но эти кривые описывают не самую естественную траекторию движения для автомобиля.

Сплайны

Сплайны [2] более точно описывает дорогу, чем предыдущие две модели, и могут очень точно аппроксимировать множество точек, но у сплайнов не очень удобно считать кривизну, по которой строится траектория.

Клотоиды

В том числе для аппроксимации разметки используют [3] клотоиду. Клотоида — кривая, у которой кривизна меняется линейно от длины. Данный вид кривой хорошо описывает повороты на дороге и удобен для вычисления кривизны, однако вычислительно сложен в построении и сглаживании.

Дуги и отрезки

Дуги достаточно хорошо описывают физическую модель дороги в виде сверху. Данная модель с некоторыми ограничениями и была выбрана в рамках данной работы.

Выбранная модель полосы

В данной работе в качестве модели полосы были выбраны две или одна кривая одного вида: дуги или прямые с ограничением на центры или параллельность соответственно. Не используется жесткая модель с концентрическими дугами или параллельными прямыми, так как из-за погрешности нахождения разметки и погрешности определения горизонта при перепроекци-

ровании к виду сверху теряется точность и выгодно использовать менее жесткую модель.

Построение полосы

Во входных данных встречается существенное количество ложно-положительных кандидатов на разметку, к примеру, блики на дороге. Помимо них на дороге встречаются соседние полосы и прочая разметка, которая нам не интересна. Было принято решение сперва провести первичную фильтрацию: были отфильтрованы кривые по кривизне, положению по оси x и по углу наклона. Поскольку радиус поворота дороги не может быть меньше 15 метров по законодательству, то таким образом мы избавляемся от части разметки, дублирующей дорожные знаки.

После первичной фильтрации все еще могут оставаться ложно-положительный кандидаты. Для их определения производится отслеживание кандидатов на разметку от кадра к кадру, в качестве метрики для сравнения используется *maximal deviation*. Чтобы не терять фрагменты разметки из-за сдвига автомобиля относительно дороги, используются данные одометрии.

После фильтрации возникает необходимость разделить кандидатов на разметку на части левой и правой полосы. Для этого выделяются наиболее вероятные начала левой и правой полосы. За начала принимаются два кандидата на разметку, находящиеся на расстоянии не меньше двух метров и находящиеся наиболее близко к камере по оси y . Эти кривые наиболее вероятно являются началами нужной полосы из-за ограничения угла обзора: видимые точки соседних полос находятся дальше от машины.

Далее мы предполагаем, что соседние фрагменты разметки находятся примерно на одной прямой. С помощью последовательностей окошек поиска в направлении последнего найденного фрагмента находим следующий кандидат на разметку. Таким образом мы получаем две группы точек, которые делим на левые и правые.

После было решено использовать информацию, извлеченную из трекинга. Не учитываем при построении модели кандидатов на разметку, которые появились только на последнем кадре и используем тех, которые стабильно появляются и ранее участвовали в построении полосы.

Далее каждое из множеств точек аппроксимируем окружностями с помощью алгоритма RANSAC. Если полученный радиус окружности превышает 2000 метров, то считаем, что на данном кадре нет поворота и дорога прямая. Апроксимация дороги прямой осуществляется с помощью метода наименьших квадратов. Смена типа кривых в модели происходит по гистерезису.

Если были найдены точки и левой и правой части полосы в достаточном количестве для построения, приводим прямые к параллельному виду, а дуги к концентрическому виду.

Чтобы не было дергания полосы между кадрами, кривые гладживаются. Для сглаживания фильтруем экспоненциальным фильтром точки, принадлежащие кривым на расстоянии 2.5 м, 13 м, 20 м, и после фильтрации повторно строим дуги или прямые. Помимо этого, фильтруется ширина полосы. Отравляем данные в фильтр после приведения кривых к концентрическому-/параллельному виду.

Для отключения отображения полосы в случае низкой уверенности в ней вычисляется confidence модели. Используется информация о количестве кандидатов, используемых в построении, которые появлялись на предыдущих кадрах хотя бы три раза, а так же оценивается, как хорошо модель ложится на кандидатов на разметку. Может понадобиться отключить отображение, если полос на данных момент нет на кадре или алгоритм выдает неправдоподобный результат по какой-либо причине.

Эксперимент

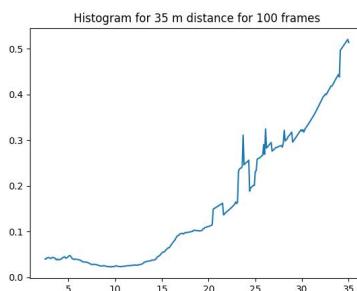


Рис. 1: Среднее расстояния до ближайшего кандидата на разметку на 100 кадрах

Для определения качества алгоритма строится гистограмма расстояний до ближайшего кандидата на разметку на разных расстояниях (см. Рис. 1, 2) от автомобиля.

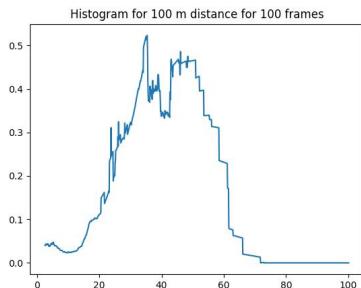


Рис. 2: Среднее расстояния до ближайшего кандидата на разметку на 100 кадрах

Заключение

В результаты была выбрана модель полосы, был реализован алгоритм ее выделения, сглаживания и вычисления confidence модели. Были произведена оценка качества алгоритма: первые 30 метров алгоритм точно строит модель: средняя ошибка не превышает 0.4 метра, тогда как ширина дорожной разметки составляет от 20 до 40 сантиметров, а дальше с увеличением расстояния точность падает.

Список литературы

- [1] Xin-yu WANG, Jian LI, Xiang-jing AN, Han-gen HE. FPGA based Curve Lane Marking Detection for ADAS. — 2017 — https://www.worldscientific.com/doi/10.1142/9789813206823_0032 (дата обращения: 13.05.2024)
- [2] Chien-Hung Yu, Kun-Lung Ku, Hung-Pang Lin. A Camera-IMU Sensor Fusion for Robust Lane Information on Lateral Control System. — 2019 — <https://ieeexplore.ieee.org/document/8951693> (дата обращения: 13.05.2024)
- [3] Yuchen Liu, Hao Cheng, Zhiqiang Li. Fused Front Lane Trajectory Estimation Based on Current ADAS Sensor Configuration. — 2020 — <https://ieeexplore.ieee.org/document/9338470> (дата обращения: 13.05.2024)

Эффективное оценивание параметров смеси распределений

Казанцев А.А., СПбГУ, Санкт-Петербург anton.a.kazancev@gmail.com,
Гориховский В.И., доцент кафедры системного программирования СПбГУ, к.ф.-м.н.,
Санкт-Петербург.

Аннотация

Оценка параметров смесей распределений встречается в большом спектре прикладных задач. Наиболее часто с решением данной проблемы сталкиваются при анализе смесей нормальных распределений. Однако бывают задачи, в которых необходимо разделять и оценивать смеси в более общем виде. Например, задержки пакетов в маршрутизаторе могут рассматриваться как распределения Вейбулла, если исключить единичные выбросы, связанные с работой сетевого оборудования. Таким образом, определение компонент смеси распределений Вейбулла позволяет строить качественные модели задержек передачи пакетов в сетях.

Среди существующих статей и библиотек присутствует большое количество узконаправленных алгоритмов для анализа распределений и их смесей. Однако в ходе анализа предметной области, не удалось найти универсальных инструментов для нахождения параметров произвольных смесей распределений.

Основная цель работы — разработка и реализация универсального метода для оценки параметров произвольных смесей непрерывных распределений.

Введение и постановка задачи

Во многих прикладных задачах, связанных с математической статистикой, возникает необходимость оценки параметров распределения случайной величины. Для решения такой задачи могут быть использованы известные для конкретного распределения оценки параметров, такие как математическое ожидание и дисперсия для нормального распределения. В том случае, если хороших известных оценок нет, можно воспользоваться математическими оптимизаторами, позволяющими находить локальные экстремумы функций от вектора параметров.

Помимо задачи об оценке параметров для одного распределения случайной величины, существует также задача оценки параметров для смеси

распределений случайной величины — комбинации нескольких распределений [1]. Такая задача возникает при наблюдении более сложных процессов, в которых участвуют случайные величины более чем из одного распределения. При оценке смеси решается ряд задач: узнать, сколько различных распределений находится в смеси, определить вид и параметры каждого распределения, узнать соотношение распределений в смеси друг к другу (априорная вероятность).

Наиболее часто с оценкой параметров смеси распределений сталкиваются при анализе смесей нормальных распределений [3]. Однако бывают задачи, в которых необходимо разделять и оценивать смеси в более общем виде. Например, смеси произвольных распределений, или смеси распределений из разных семейств.

Компания Huawei в октябре 2023 года опубликовала интернет-проект, посвящённый предсказанию задержек отправки пакетов через сеть, по наблюдаемым отправителем задержкам пакетов [7]. Задержки пакетов в маршрутизаторе могут рассматриваться с помощью распределения Вейбулла [2, 5], если исключить единичные выбросы, связанные с работой сетевого оборудования. Таким образом, определение компонент смеси распределений Вейбулла позволяет строить качественные модели задержек передачи пакетов в сетях.

Среди существующих статей и библиотек присутствует большое количество узконаправленных алгоритмов для анализа распределений и их смесей. Однако в ходе анализа предметной области не удалось найти универсальных инструментов для нахождения параметров произвольных смесей распределений.

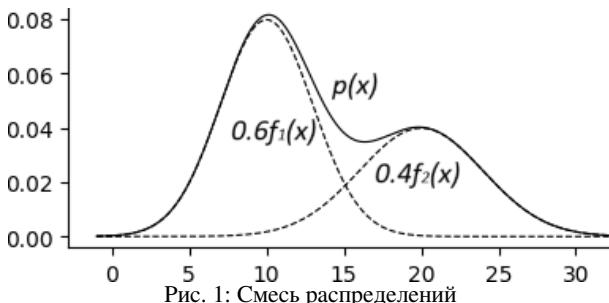
Цель работы — создание библиотеки для оценки параметров произвольных смесей непрерывных распределений случайных величин, что позволит решать большой спектр прикладных задач.

Для осуществления этой цели были сформулированы следующие **задачи**.

- Создать алгоритм для оценки параметров смесей распределений на основе ЕМ-алгоритма с поддержкой использования различных методов математической оптимизации.
- Спроектировать архитектуру библиотеки, реализующую данный алгоритм.
- Выполнить и опубликовать реализацию спроектированной библиотеки.
- Выполнить эксперименты для анализа работы алгоритма при оценке параметров смесей распределений с различными параметрами алгоритма.

Обзор

Смесь распределений



Плотность смеси распределений задается следующим образом:

$$p(x|\Omega, F, \Theta) = \sum_{j=1}^k \omega_j f_j(x|\theta_j),$$

где k — это количество компонент смеси, $\omega_j \geq 0, \sum_{j=1}^k \omega_j = 1$ — априорная вероятность j компоненты смеси, $f_j(x, \theta_j), \theta_j \in \Theta_{f_j}$ — функция правдоподобия j компоненты смеси; $\Omega = [\omega_1, \dots, \omega_k]$, $F = [f_1, \dots, f_k]$, $\Theta = [\theta_1, \dots, \theta_k]$ — параметры смеси.

На рис. 1 показана смесь, состоящая из двух распределений Гаусса, с априорными вероятностями 0.4 и 0.6.

Математическая постановка задачи

По заданной выборке X^m случайных независимых наблюдений из смеси распределений $p(x|\Omega, F, \Theta)$ оценить параметры Ω, F, Θ и количество компонент смеси k .

В данной работе рассматривается упрощенная версия задачи, в которой параметры F, k смеси известны. Полученный алгоритм в дальнейшем может использоваться внутри алгоритма, решающего полную задачу об оценке параметров смеси распределений.

EM-алгоритм

EM-алгоритм [6, 4] является итеративным алгоритмом для нахождения локального максимума логарифма функции максимального правдоподобия статистической модели. Он применяется в случае, когда модель зависит от некоторых скрытых параметров.

Алгоритм состоит из двух шагов:

- expectation (E-шаг): вычисление оценки функции максимального правдоподобия модели, при этом скрытые параметры рассматриваются как наблюдаемые.
- maximization (M-шаг): подбор скрытых параметров для того что бы максимизировать оценку, найденную на E-шаге.

Для начала работы EM-алгоритма необходимо стартовое предположение о скрытых параметрах, от которого напрямую зависит работа алгоритма.

```

function EM_step(X, Ω, F, Θ)
    \\E-step
    for j ∈ 1..k do
        for i ∈ 1..m do
             $H_{ij} \leftarrow \frac{\omega_j f_j(X_i | \theta_j)}{\sum_{s=1}^k \omega_s f_s(X_i | \theta_s)}$ 
    \\M-step
    for j ∈ 1..k do
         $\omega'_j \leftarrow \frac{1}{m} \sum_{i=1}^m H_{ij}$ 
         $\theta'_j \leftarrow \operatorname{argmax}_{\gamma \in \Theta_{f_j}} (\sum_{i=1}^m H_{ij} \ln F_j(X_i, \gamma))$ 
    return Ω', Θ'

```

Алгоритм для оценки параметров смесей распределений

В основе используемого в библиотеке алгоритма лежит описанная ранее версия ЕМ-алгоритма в общем виде с добавлением функции останова и проверки рассматриваемых распределений на их корректность.

```
function EM_solve( $X, \Omega, F, \Theta$ )
     $\Omega', F', \Theta' \leftarrow \Omega, F, \Theta$ 
    repeat
         $\Omega, F, \Theta \leftarrow \Omega', F', \Theta'$ 
         $\Omega', \Theta' \leftarrow EM\_step(X, \Omega, F, \Theta)$ 
         $F' \leftarrow [F_j \in F : valid(\Omega'_j, F_j, \Theta'_j)]$ 
    until  $breakpoint(\Omega, F, \Theta, \Omega', F', \Theta')$ 
    return  $F', \Omega', \Theta'$ 
```

- $valid(\omega, f, \theta)$ — проверка, является ли распределение с соответствующими параметрами корректным
- $breakpoint(\Omega, F, \Theta, \Omega', F', \Theta')$ — функция точки останова

Критерии останова

Так как ЕМ-алгоритм итеративный, можно выделить большое количество различных критериев останова. Примеры возможных критериев останова:

- время исполнения:
 - максимальное количества шагов;
 - максимальное времени исполнения;
- относительная точность — изменение параметров;
- абсолютная точность:
 - логарифм максимального правдоподобия;
 - разделение выборки на рабочую и тестовую.

«Плохие» распределения

В связи с использованием оптимизатора в алгоритме, параметры одного или нескольких распределений в выборке могут стать экстремально большими. Данная ситуация может возникнуть по ряду причин, среди которых слишком большой параметр k и плохое стартовое предположение. Так как распределение в выборке имеет параметр ω (априорная вероятность), с точки зрения алгоритма ошибок нет, так как при экстремально больших ложных параметрах ω стремится к нулю. Наличие в выборке «плохого» распределения может привести к значительному снижению скорости работы алгоритма, или к его остановке.

После очередного М-шага алгоритма следует проводить проверку корректности полученных распределений. В случае нахождения «плохого распределения» следует отбросить его и не использовать в дальнейшей работе алгоритма, сообщив что стартовое предположение, приведшее к возникновению этого распределения, ложно.

Таким образом, к возможным критериям «плохого» распределения можно отнести малую априорную вероятность или экстремально большие значения параметров.

Проектирование и реализация библиотеки

Была спроектирована библиотека, позволяющая решать задачу оценки параметров произвольных смесей непрерывных или дискретных распределений [10].

Библиотека была спроектирована с требованиями к универсальности и простоте модификаций.

Архитектура библиотеки

Основной класс библиотеки позволяет применять описанный выше ЕМ-алгоритм к произвольным распределениям, описанным с помощью содержащихся в библиотеке классов обёрток. При его инициализации необходимо в качестве параметров передать оптимизатор, класс реализующий условие останова и класс определяющий, является ли распределение из смеси корректным.

В соответствии с требованиями ЕМ-алгоритма и используемого в нём оптимизатора, сформулируем следующие требования к описанию модели распределения:

- плотность;
- логарифм плотности;
- логарифм частных производных плотности по каждому параметру (дополнительно для работы с оптимизаторами второго порядка).

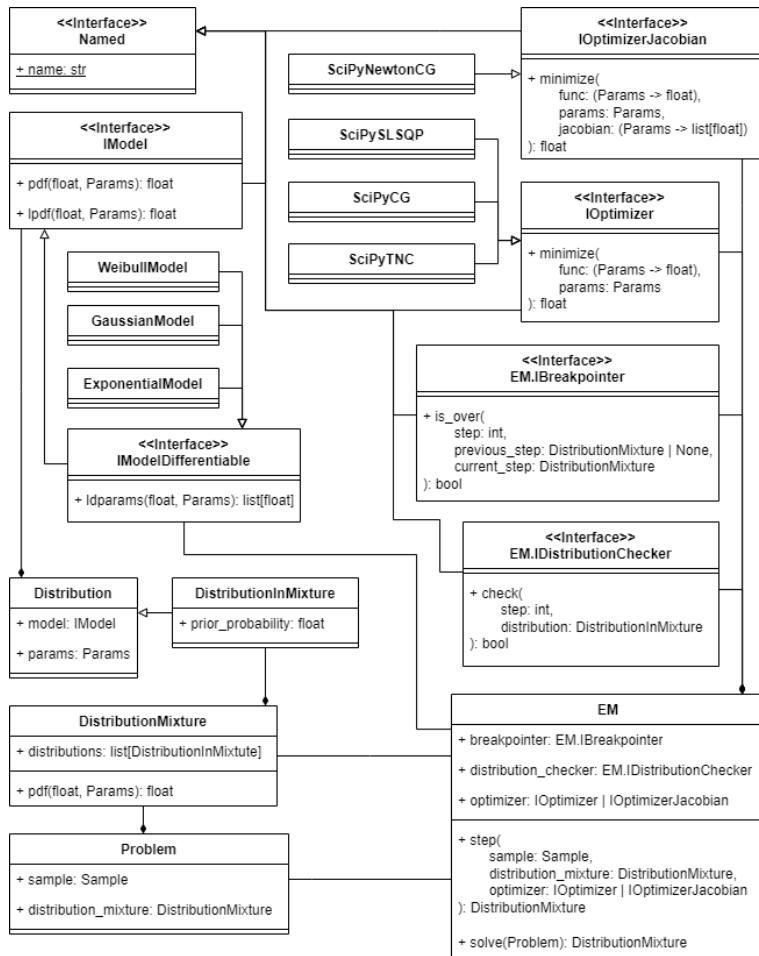


Рис. 2: UML диаграмма классов библиотеки

Эксперименты

Дизайн эксперимента

Характеристики оборудования

- Ноутбук: HP Probook 440 G5, подключенный к сети питания
- Процессор: Intel Core i5-8250U
- Оперативная память: DDR4 16Гб
- Версия ОС: Ubuntu

Рассматриваемые распределения

Так как в ходе работы используемого алгоритма подразумевается использование функций оптимизации (максимизации), плотность рассматриваемых распределений должна быть всюду дифференцируемой по её параметрам. Для того что бы обеспечить такое свойство распределению, параметры которого не могут быть отрицательными, применяется замена параметра на его экспоненту.

Распределение Вейбулла Наиболее интересное для изучения в данной работе распределение, так как его параметры k и λ трудно выразить через выборку.

Плотность распределения Вейбулла:

$$f(x|k, \lambda) = \frac{k}{\lambda} \left(\frac{x}{\lambda}\right)^{k-1} e^{-\left(\frac{x}{\lambda}\right)^k}, k > 0, \lambda > 0, x \geq 0$$

Распределение Гаусса

$$f(x|m, \sigma) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(n-x)^2}{2\sigma^2}}, m \in \mathbb{R}, \sigma >= 0, x \in \mathbb{R}$$

Экспоненциальное распределение

$$f(x|\lambda) = \lambda e^{(-\lambda x)}, \lambda > 0, x >= 0$$

Эксперимент со смесями одного семейства распределений

Условия эксперимента

- Смеси распределений Вейбулла, Гаусса, экспоненциальные
- Оптимизаторы из пакета SciPy [9]: CG, NewtonCG, SLSQP, TNC
- Количество компонент от 1 до 3
- Генерируются 32 выборки из 2048 случайных независимых наблюдений для каждого количества компонент и типов распределений, откуда в дальнейшем случайно выбираются выборки меньших размеров.
- Размеры компонент выборки: 50, 100, 200, 500, 1000, одинаковые для одного теста
- По 2 выборки меньшего размера из каждой большой выборки
- По 8 независимых тестов со случайными стартовыми параметрами
- Максимальное количество шагов алгоритма 16
- Минимальная априорная вероятность распределения в выборке 0.1%

Обоснование Данный эксперимент направлен на то, что бы понять, как работает алгоритм и какие ситуации могут возникнуть во время работы. А также сравнить работу наиболее интересных оптимизаторов.

Были взяты одинаковые размеры компонент выборки для упрощения анализа результатов.

На практике хочется получить не очень точный, но быстрый алгоритм, поэтому размеры компонент начинаются с 50. Выборки с компонентами размера 1000 нужны для понимания того, как сильно увеличится время работы алгоритма и насколько точнее он будет работать.

Метрики

- скорость схождения;
- усреднённый логарифм максимального правдоподобия;

$$NLL = \frac{1}{m} \sum_{x \in X^m} \ln p(x|\Omega, F, \Theta)$$

- шанс ошибки алгоритма: алгоритм не смог найти ни одного распределения или остановил работу из-за экстремально больших параметров смеси.

Результаты

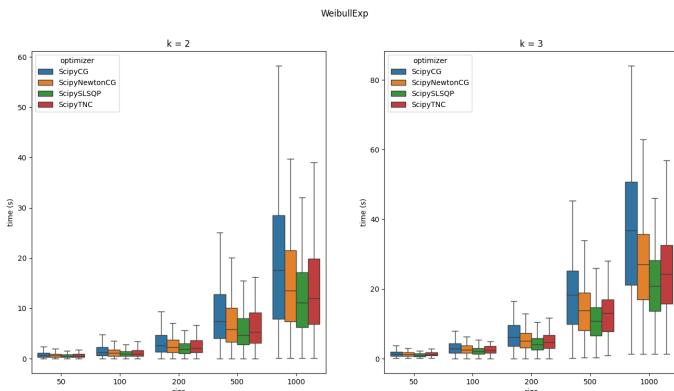


Рис. 3: Время работы алгоритма при оценке смесей Вейбулла

На рис. 3 изображён график зависимости времени исполнения алгоритма от размера выборки, количества распределений в смеси и оптимизатора для смесей Вейбулла. Чем больше размер выборки, тем больше времени необходимо алгоритму, что бы сойтись. Так же видно, что оптимизаторы работают с разной скоростью. Самым медленным оказался оптимизатор CG (метод со-пряженных градиентов). Самым же быстрым — SLSQP (последовательное квадратичное программирование).

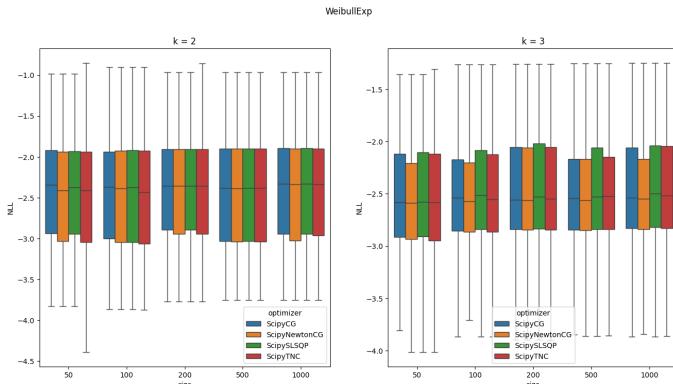


Рис. 4: NLL алгоритма при оценке смесей Вейбулла

На рис. 4 видно, что различные оптимизаторы практически не отличаются в точности для задачи оценки параметров смесей распределений Вейбулла.

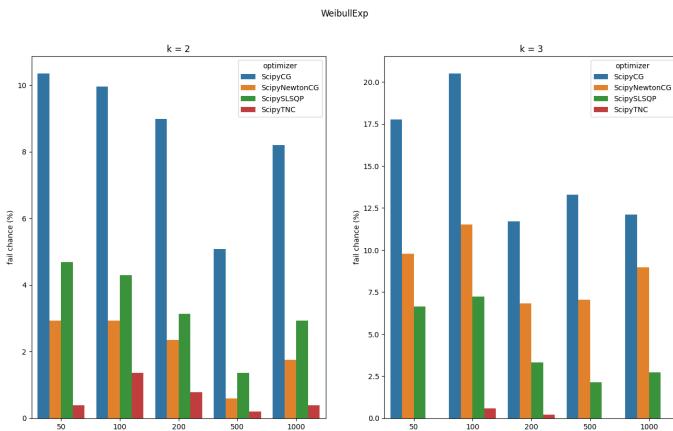


Рис. 5: Шанс ошибки алгоритма при оценке смесей Вейбулла

На рис. 5 видно, что шанс ошибки алгоритма зависит от оптимизатора. Для смесей Вейбулла самым стойким оказался оптимизатор TNC (Truncated Newton method).

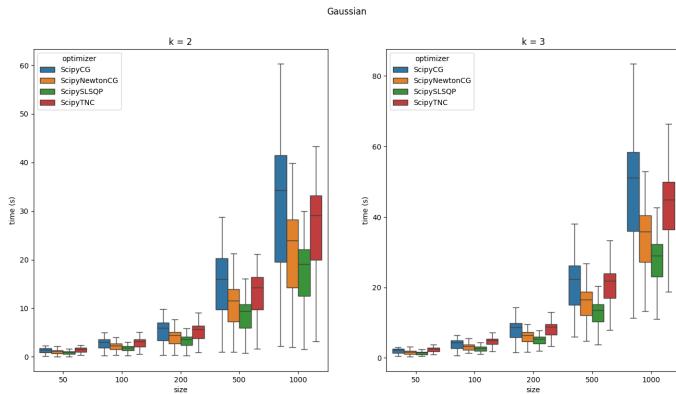


Рис. 6: Время работы алгоритма при оценке смесей Гаусса

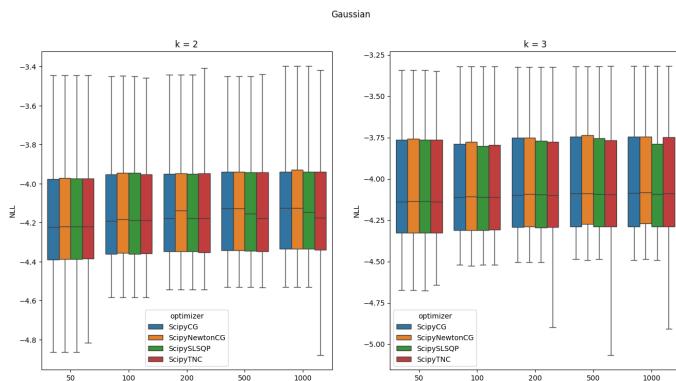


Рис. 7: NLL алгоритма при оценке смесей Гаусса

Для смесей распределений Гаусса время работы и точность (Рис. 6, Рис. 7) зависят от оптимизатора практически так же как и для смесей Вейбулла.

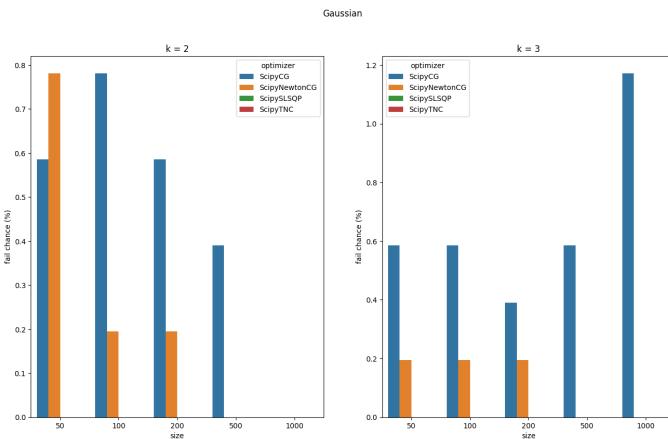


Рис. 8: Шанс ошибки алгоритма при оценке смесей Гаусса

Шанс ошибки алгоритма при оценке параметров смесей Гаусса (Рис. 8), в отличие от Вейбулла, минимален не только у оптимизатора TNC, но и у SLSQP. Таким образом для смесей Гаусса оптимизатор SLSQP является лучшим по времени работы и по шансу ошибки.

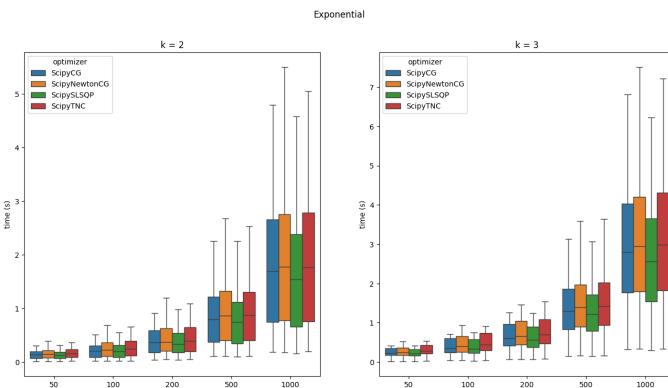


Рис. 9: Время работы алгоритма при оценке смесей экспоненциальных распределений

Как видно из рис. 9, при оценке параметров смеси экспоненциальных распределений разница между рассмотренными оптимизаторами не столь велика, как для смесей Вейбулла и Гаусса.

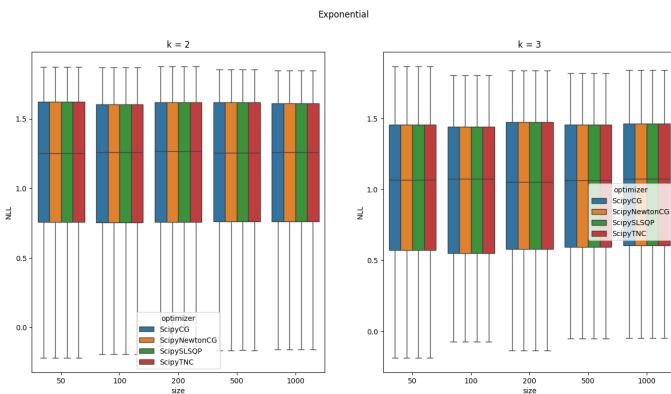


Рис. 10: NLL алгоритма при оценке смесей экспоненциальных распределений

Так же как и для смесей Вейбулла и Гаусса, точность алгоритма при оценке параметров смеси экспоненциальных распределений (Рис. 10) не сильно зависит от оптимизатора.

При оценке параметров смесей экспоненциальных распределений ни один из оптимизаторов не ошибся.

По графикам можно сделать следующие наблюдения:

- Оценка параметров смеси экспонент быстрее и точнее, чем Гаусса и Вейбулла, что вероятно связано с тем, что экспоненциальное распределение зависит от одного параметра.
- Различные оптимизаторы работают с разной скоростью. Самый быстрый оптимизатор среди рассмотренных: SLSQP.
- У разных оптимизаторов разная устойчивость к ошибкам. Самыми устойчивыми оказались оптимизаторы SLSQP и TNC.
- Точность оценки параметров практически не зависит от оптимизатора.

Заключение

- Был создан алгоритм для решения поставленной задачи (на основе ЕМ-алгоритма в общем виде). Работа полученного алгоритма зависит от следующих параметров:
 - метод математической оптимизации;
 - условие остановки;
 - условие корректности рассматриваемых распределений в смеси.
- Спроектирована архитектура библиотеки, реализующей предложенный алгоритм, которая отвечает требованиям по универсальности и расширяемости, позволяет работать со смесями распределений из различных семейств и дополнять библиотеку произвольными моделями распределений и различными реализациями ключевых параметров алгоритма.
- Библиотека реализована на языке Python.
- Выполнено экспериментальное исследование, получены следующие результаты/выводы:
 - доказана корректность разработанного алгоритма;
 - выявлено, что с увеличением количества распределений в смеси растёт время работы и падает точность алгоритма, а с увеличением размера выборки растёт время работы и точность алгоритма;
 - показано, что методы оптимизации обладают разными достоинствами/недостатками и для эффективного решения задачи они могут быть использованы в комбинации друг с другом.

Список литературы

- [1] Bruce G. Lindsay: Mixture Models: Theory, Geometry and Applications. NSF-CBMS Regional Conference Series in Probability and Statistics. 1995. Hayward, CA, USA. <https://www.jstor.org/stable/4153184>
- [2] Elmahdy Emad E., Aboutahoun Abdallah W.: A new approach for parameter estimation of finite Weibull mixture distributions for reliability modeling. Applied Mathematical Modelling. 2013. c. 1800–1810. <https://www.sciencedirect.com/science/article/pii/S0307904X12002545>
- [3] Wardatul Jannah, Dewi R.S. Saputro: Parameter estimation of Gaussian mixture models (GMM) with expectation maximization (EM) algorithm. Conference Proceedings 2566, 040002. 2022. https://web.archive.org/web/20221201095213id_/https://aip.scitation.org/doi/pdf/10.1063/5.0117119
- [4] Jason Brownlee: A Gentle Introduction to Expectation-Maximization (EM Algorithm). 2020. <https://machinelearningmastery.com/expectation-maximization-em-algorithm>
- [5] И.М. Макуха: Оценка параметров смеси распределений Вейбулла, выпускная квалификационная работа Санкт-Петербургского государственного университета. 2023. https://se.math.spbu.ru/thesis/texts/Makuha_I1'ja_Mihajlovich_Bachelor_Thesis_2023_text.pdf
- [6] J.W. Davenport, J.C. Bezdek, R.J. Hataway: Parameter estimation for finite mixture distributions. Mathematics and Computer Science Department. 1988. Southern College, Georgia. <https://core.ac.uk/download/pdf/82096497.pdf>
- [7] Dmitriy Moskvitin, Evgeny Onegin, Rachel Huang, Hanlin Luo, Qichang Chen: Forward Erasure Correction for Short-Message Delay-Sensitive QUIC Connections. 2023. <https://datatracker.ietf.org/doc/draft-dmoskvitin-quic-short-message-fec>
- [8] Matthew Reid: Reliability — a Python library for reliability engineering. 2022. <https://doi.org/10.5281/ZENODO.3938000>
- [9] SciPy — an open-source software for mathematics, science, and engineering. <https://docs.scipy.org/doc/scipy/>
- [10] Реализация библиотеки. <https://github.com/toxakaz/EM-algo>

Оптимизация алгоритма контекстно-свободной достижимости, основанного на операциях линейной алгебры

Муравьев И.В., СПбГУ, Санкт-Петербург, muraviovilya@gmail.com

Аннотация

Различные задачи статического анализа кода и анализа RDF-графов сводятся к задаче контекстно-свободной (КС) достижимости. Одним из многообещающих способов решения задачи КС-достижимости для встречающихся на практике больших графов является решение задачи в терминах операций линейной алгебры с разреженными матрицами, так как эти операции крайне эффективно выполняются на современном оборудовании. В работе предложены, реализованы и протестированы шесть оптимизаций матричного алгоритма КС-достижимости, основанных на свойствах полукольца КС-достижимости, специфике представления КС-языков и особенностях популярной библиотеки линейной алгебры SuiteSparse:GraphBLAS. Экспериментально установлено, что предложенные оптимизации позволяют в подавляющем большинстве случаев на порядки ускорить матричный решатель КС-достижимости и обогнать другие передовые универсальные КС-решатели.

Введение

Задача контекстно-свободной (КС) достижимости [1] — это задача поиска в рёберно-меченнем графе пар вершин, соединённых таким путём, что метки рёбер вдоль этого пути формируют слово из заданного контекстно-свободного языка. Ряд вычислительно затратных задач сводится к задаче КС-достижимости, в частности: задача анализа потока значений (англ. *value-flow analysis*) [2], задача анализа псевдонимов (англ. *alias analysis*) [3], задача анализа указателей (англ. *points-to analysis*) [4] и задача поиска концептов одного уровня в RDF-графах (англ. *same layer of the hierarchy of RDF graphs*) [5].

Существует множество алгоритмов для решения задачи КС-достижимости [6, 7, 8, 2, 9]. Одним из перспективных универсальных подходов является матричный алгоритм Рустама Азимова [6]. Данный алгоритм стремится достичь высокой производительности за счёт решения задачи в терминах операций с разреженными матрицами, эффективно выполняемыми на современном параллельном аппаратном обеспечении, и

часто превосходит передовые аналоги по скорости работы [10]. Тем не менее, несмотря на эффективность матричных операций, в ряде случаев матричный алгоритм, как и другие инструменты, работает крайне долго [11, 12], в частности когда требуется найти пути с глубокими деревьями вывода и/или используются КС-грамматики с большим количеством правил вывода.

Следовательно, одним из перспективных способов ускорить решение упомянутых вычислительно-затратных задач является оптимизация матричного алгоритма КС-достижимости, чему и посвящена данная работа.

Терминология

Определение 1. (Решение задачи КС-достижимости) Пусть G — это рёберно-меченный граф с вершинами V , а Gr — КС-грамматика со стартовым нетерминалом S . Тогда *решением задачи КС-достижимости* называют множество $\{\langle v_i, v_j \rangle \in V^2 \mid \exists \text{ путь из } v_i \text{ в } v_j, \text{ выводимый из нетерминала } S\}$.

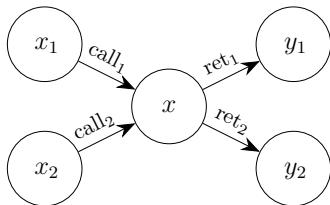


Рис. 1: Граф для анализа потока знаний в программе на рис. 2.

```

def identity(x):
    return x

def main():
    y1 = identity(x1)
    y2 = identity(x2)
  
```

Рис. 2: Пример анализируемой Python-программы.

Пример 1. Пусть G — это граф на рис. 1, а Gr — КС-грамматика со следующими правилами вывода:

$$\begin{aligned} \forall i \in \{1, 2\} : S &\rightarrow call_i \ ret_i; \\ \forall i \in \{1, 2\} : S &\rightarrow call_i \ S \ ret_i \ S. \end{aligned}$$

Тогда решением задачи КС-достижимости для G и Gr является множество $\{\langle x_1, y_1 \rangle, \langle x_2, y_2 \rangle\}$, так как $x_1 \xrightarrow{\text{call}_1 \ \text{ret}_1} y_1$ и $x_2 \xrightarrow{\text{call}_2 \ \text{ret}_2} y_2$.

Оптимизации

В ходе работы путём профилирования и анализа временной сложности последовательно выявлялись и ускорялись наиболее узкие места матричного

алгоритма КС-достижимости Рустама Азимова [6]. В результате этого процесса были предложены следующие оптимизации:

- инкрементальное вычисление произведений матриц (переиспользование результатов предыдущих итераций алгоритма);
- быстрое выполнение операций с нулевыми матрицами без вызова функций библиотеки линейной алгебры;
- динамический выбор между строчным и столбцовыми форматами представления матриц с возможностью хранения частей одной матрицы над полукольцом КС-достижимости в разных форматах;
- динамический выбор между символьными и конкретными вычислениями (иногда целесообразнее не вычислять значение матрицы, а запомнить выражение, позволяющее его вычислить, чтобы потом подставлять это выражение при каждом использовании матрицы и производить упрощения);
- использование блочных матриц для эффективной работы с встречающимися на практике КС-грамматиками с большим количеством однотипных правил вывода, то есть КС-грамматиками, в которых, как в КС-грамматике из примера 1, есть правила, повторяющиеся для всех допустимых значений i ;
- выявление эмпирическим путём эквивалентных КС-грамматик, задающих тот же КС-язык, что и входная КС-грамматика, но требующих меньшего объёма вычислений для решения задачи.

Реализация

Предложенные оптимизации были реализованы в разработанном в рамках работы матричном КС-решателе FastMatrixCFPQ¹. Решатель был разработан на языке программирования Python с использованием библиотеки разреженной линейной алгебры SuiteSparse:GraphBLAS² и состоит из четырёх компонентов верхнего уровня со следующими зонами ответственности:

- `cfpq_matrix` — оптимизации на уровне отдельных матриц, а также ускорение сборки мусора (матриц, которые перестали использоваться);

¹Разработанный в рамках работы решатель задачи КС-достижимости — https://github.com/FormalLanguageConstrainedPathQuerying/CFPQ_PyAlgo/tree/murav/optimize-matrix.

²Библиотека для параллельной работы с графами на языке разреженной линейной алгебры, SuiteSparse:GraphBLAS — <https://github.com/DrTimothyAldenDavis/GraphBLAS>.

- `cfpq_model` — представление графов и КС-языков;
- `cfpq_algo` — реализации матричного алгоритма КС-достижимости;
- `cfpq_cli` — интерфейс командной строки.

Большую часть предложенных оптимизаций удалось реализовать в виде обёрток, которые можно добавлять и убирать, что позволило покрыть интеграционными тестами все 64 комбинации предложенных оптимизаций.

Помимо КС-решателя, в рамках работы была разработана система, автоматизирующая экспериментальное сравнение КС-решателей³. Эта система была проинтегрирована с шестью передовыми КС-решателями [4, 2, 6, 9, 13, 14] и поддерживает настройку параметров экспериментов, прерывание и возобновление экспериментов, а также визуализацию результатов.

Наконец, на языке программирования Kotlin был разработан вспомогательный проект⁴, отвечающий за построение дополнительных наборов входных данных по Java-программам с помощью анализа трёхадресных инструкций, в которые библиотека JacoDB⁵ преобразует JVM-байткод.

Эксперимент

Перед проведением экспериментального исследования были сформулированы два исследовательских вопроса.

RQ1. Даёт ли FastMatrixCFPQ правильные ответы?

RQ2. Когда FastMatrixCFPQ работает быстрее аналогов и/или использует меньше оперативной памяти?

Для ответа на эти вопросы было проведено экспериментальное сравнение FastMatrixCFPQ с передовыми аналогами: MatrixCFPQ [6], PEARL 2023 [9], POCR 2022 [2], KotGLL 2023 [14], Graspan [13] и Gigascale [4]. Входные данные были взяты из наборов данных CFPQ_Data⁶ и CPU 17⁷, на которых замерили свою производительность перечисленные аналоги. Входные данные

³Разработанная в рамках работы система автоматизации экспериментов — https://github.com/FormalLanguageConstrainedPathQuerying/CFPQ_PyAlgo/tree/murav/optimize-matrix/cfpq_eval.

⁴Разработанный в рамках работы проект для построения графов по Java-программам — https://github.com/FormalLanguageConstrainedPathQuerying/CFPQ_JavaGraphMiner.

⁵Библиотека для анализа JVM-байткода, JacoDB — <https://jacodb.org>.

⁶Набор данных CFPQ_Data, графы и КС-грамматики для оценки производительности КС-решателей — https://github.com/FormalLanguageConstrainedPathQuerying/CFPQ_Data.

⁷Набор данных CPU 17, графы программ из набора данных SPEC 2017 C/C++ — <https://github.com/kisslune/CPU17-graphs>.

были дополнены графами для анализа указателей, построенными в рамках работы по популярным Java-библиотекам.

Все рассмотренные наборы данных в совокупности содержат более 50 графов. Каждый граф относится к одной из пяти прикладных задач, требующих использования различных КС-грамматик. На рис. 3 и 4 каждая задача представлена одним графом. Чтобы выборка представленных графов была максимально непредвзятой, для каждой задачи выбран тот график, который FastMatrixCFPQ анализировал дольше остальных графов для той же задачи.

Для замеров производительности использовались центральный процессор AMD Ryzen 9 7900X и 128 ГБ оперативной памяти DDR5. Программное окружение было зафиксировано с помощью Docker-образа⁸.

RQ1: правильность ответов. Для всех рассмотренных входных данных FastMatrixCFPQ получил ответы, совпавшие с ответами других корректно работающих КС-решателей.

RQ2: производительность. Как показано на рис. 3, FastMatrixCFPQ для показанных графов обгоняет все рассмотренные аналоги, кроме Gigascale [4]. Это объясняется тем, что Gigascale — это специализированный инструмент, оптимизированный для работы с одним конкретным КС-языком. Что касается использования оперативной памяти, то, как показано на рис. 4, для FastMatrixCFPQ обычно требуется меньше памяти, чем для аналогов, но есть исключения, в частности, для графа «taxon_h» меньше памяти требуется инструменту Graspan [13], а для графа «perlb_aa» — инструменту MatrixCFPQ [6]. Однако стоит отметить, что в данных случаях соответствующие инструменты работают более чем в 8 раз дольше (см. рис. 3).

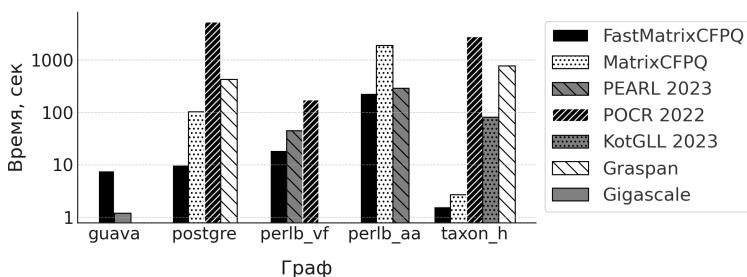


Рис. 3: Время работы КС-решателей, выборочное стандартное отклонение по результатам пяти замеров в пределах 7%. Отсутствие столбца обозначает превышение ограничения по времени в 10 000 секунд, ограничения по памяти в 128 ГБ или несовместимость инструмента с задачей.

⁸Репозиторий с Docker-образом для проведения экспериментального сравнения решателей КС-достижимости — https://hub.docker.com/r/cfpq/py_algo_eval.

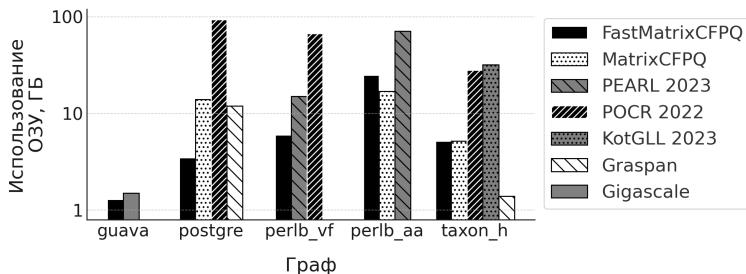


Рис. 4: Использование ОЗУ КС-решателями, выборочное стандартное отклонение по результатам пяти замеров в пределах 4%. Обозначения аналогичны рис. 3.

Заключение

В работе были предложены шесть ортогональных оптимизаций матричного алгоритма КС-достижимости, позволившие обогнать передовые универсальные КС-решатели. Тем не менее, предложенных оптимизаций оказалось недостаточно, чтобы обогнать инструмент Gigascale [4], специализированный для одного конкретного КС-языка, из чего можно сделать вывод, что перспективным направлением дальнейшего развития может стать обобщение подходов Gigascale [4] для произвольных КС-языков.

Список литературы

- [1] Yannakakis M. Graph-Theoretic Methods in Database Theory // Proc. of the Ninth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems (PODS '90). — 1990. — P. 230–242. <https://doi.org/10.1145/298514.298576>
- [2] Lei Y., et al. Taming Transitive Redundancy for Context-Free Language Reachability // Proc. ACM Program. Lang. — 2022. — Vol. 6, no. OOPSLA2, art. no. 180. <https://doi.org/10.1145/3563343>
- [3] Zheng X., Rugina R. Demand-Driven Alias Analysis for C // SIGPLAN Not. — 2008. — P. 197–208. <https://doi.org/10.1145/1328897.1328464>
- [4] Dietrich J., et al. Giga-Scale Exhaustive Points-to Analysis for Java in under a Minute // Proc. of the 2015 ACM SIGPLAN International Conf. on Object-Oriented Programming, Systems, Languages, and Applications (OOPSLA 2015). — 2015. — P. 535–551. <https://doi.org/10.1145/2814270.2814307>

- [5] Zhang X., et al. Context-Free Path Queries on RDF Graphs // The Semantic Web – ISWC 2016. — 2016. — P. 632–648. https://link.springer.com/chapter/10.1007/978-3-319-46523-4_38
- [6] Azimov R., Grigorev S. Context-Free Path Querying by Matrix Multiplication // Proc. of the 1st ACM SIGMOD Joint International Workshop on Graph Data Management Experiences & Systems and Network Data Analytics (GRADES-NDA '18). — 2018. — Art. no. 5. <https://doi.org/10.1145/3210259.3210264>
- [7] Medeiros C., et al. Costa U. Efficient Evaluation of Context-Free Path Queries for Graph Databases // Proc. of the 33rd Annual ACM Symposium on Applied Computing (SAC '18). — 2018. — P. 1230–1237. <https://doi.org/10.1145/3167132.3167265>
- [8] Orachev E., et al. Context-Free Path Querying by Kronecker Product // Advances in Databases and Information Systems: 24th Europ. Conf., ADBIS 2020 — 2020. — P. 49–59. https://doi.org/10.1007/978-3-030-54832-2_6
- [9] Shi C., et al. Two Birds with One Stone: Multi-Derivation for Fast Context-Free Language Reachability Analysis // 2023 38th IEEE/ACM International Conference on Automated Software Engineering (ASE). — 2023. — P. 624–636. <https://doi.ieeecomputersociety.org/10.1109/ASE56229.2023.00118>
- [10] Azimov R. Context-Free Path Querying Using Linear Algebra. PhD thesis, St. Petersburg State University, 2022. https://disser.spbu.ru/files/2022/disser_azimov.pdf
- [11] Kutuev V. Experimental investigation of context-free-language reachability algorithms as applied to static code analysis. Master's thesis, St. Petersburg State University, 2023. <http://hdl.handle.net/11701/42628>
- [12] Kuijpers J., et al. An Experimental Study of Context-Free Path Query Evaluation Methods // Proc. of the 31st International Conference on Scientific and Statistical Database Management (SSDBM '19). — 2019. — P. 121–132. <https://doi.org/10.1145/3335783.3335791>
- [13] Wang K., et al. Graspan: A Single-Machine Disk-Based Graph System for Interprocedural Static Analyses of Large-Scale Systems Code // SIGPLAN Not. — 2017. — Vol. 52, no. 4. — P. 389–404. <https://doi.org/10.1145/3093336.3037744>
- [14] Abzalov V. Implementation and experimental investigation of the GLL parser based on a recursive automaton. Bachelor's thesis, St. Petersburg State University, 2023. <http://hdl.handle.net/11701/42730>

Разработка инфраструктуры для обучения искусственных нейронных сетей выбору оптимального пути для символьного исполнения

Нигматулин М.В., СПбГУ, Санкт-Петербург, mvnigma@gmail.com,
Чистякова А.А., СПбГУ, Санкт-Петербург, chi.vinny0702@gmail.com,
Григорьев С.В., СПбГУ, Санкт-Петербург, rsdpisuy@gmail.com

Аннотация

В данной работе представлено описание инфраструктуры для обучения графовых нейронных сетей выбору оптимального пути для символьного исполнения. Использование графовых нейронных сетей позволяет учитывать структуру программы, что в перспективе может улучшить процесс генерации тестов.

Введение

Множество инструментов для генерации тестов [11, 7, 2, 3] используют технику, позволяющую исследовать все возможные пути (ветви) исполнения программы путем использования символьных данных вместо реальных — символьное выполнение [9]. Однако у этой техники есть проблема экспоненциального увеличения числа путей (взрыва путей) во время исследования программы [4], что значительно увеличивает время генерации тестов.

Одним из способов решения проблемы взрыва путей является оптимизация выбора путей исполнения. Выбирая более оптимальные пути, можно сократить количество исследуемых состояний исполнения.

Одним из подходов к разработке стратегий выбора оптимального пути является применение машинного обучения для автоматического вывода правил выбора путей. Этот подход основывается на анализе данных об особенностях состояний исполнения [8, 5].

Так как множество состояний и путей между ними представляют собой граф, называемый также графом исполнения [6], новые данные для анализа могут быть предоставлены графовыми нейронными сетями [10] (англ. *Graph Neural Networks, GNN*). Графовые нейронные сети могут улавливать зависимости между вершинами в графах, что можно использовать для разработки стратегии выбора оптимальных путей в графе исполнения, учитывающей не только особенности состояний, но и взаимосвязи между ними.

В данной работе описывается инфраструктура для обучения графовых нейронных сетей выбору оптимального пути исполнения.

Цель работы

Цель работы — реализация инфраструктуры для обучения графовых нейронных сетей выбору оптимальных путей на основе структуры графа программы.

Для достижения этой цели необходимо создать модуль для обучения графовых нейронных сетей выбору оптимального пути для символьного исполнения, разработать протокол взаимодействия с различными символьными машинами. Дополнительно необходимо разработать модули для обеспечения возможности интеграции обученных нейронных сетей с символьными машинами и сравнения стратегий выбора пути.

Обзор

Инфраструктурные решения для машинного обучения уже существуют и успешно применяются в области создания стратегий выбора оптимального пути. Рассмотрим их с точки зрения используемых символьных машин и алгоритмов машинного обучения.

ParaDySE. В данном подходе авторы представляют решение для автоматической генерации эвристических стратегий выбора путей для символьного исполнения. В качестве символьной машины ParaDySE использует KLEE [3] и CREST [2]. В процессе символьного исполнения программы ParaDySE оптимизирует функцию выбора состояния путем подбора весов вектора признаков состояний исполнения (например, находится ли ветвь в цикле), составляющих оптимальный эвристический алгоритм для конкретной программы [5].

Q-KLEE. В данном подходе авторы используют Q-learning [12] — метод машинного обучения с подкреплением для обучения выбору оптимального пути, статический анализ программы для определения инструкций, которые Q-learning будет оценивать наградой, и KLEE в качестве символьной машины [13].

LEARCH. Еще один подход с использованием машинного обучения — LEARCH. Основной компонент LEARCH — модель машинного обучения, оценивающая награду за выбор состояния. Для каждого состояния вычисляется вектор признаков, который затем подается в модель машинного обучения для оценки [8]. Для создания набора данных для обучения и оценки моделей, LEARCH использует символьную машину KLEE.

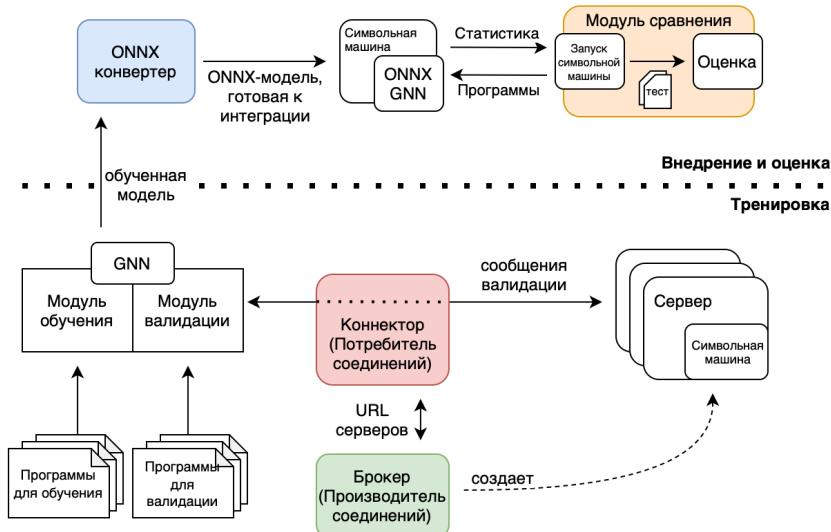


Рис. 1: Схема компонентов

Выводы

Хотя существующие подходы используют машинное обучение для создания оптимального алгоритма выбора пути для символьного исполнения, рассмотренные подходы не предполагают использование графовых нейронных сетей и реализованы на базе конкретных символьных машин. Таким образом, возникает потребность в разработке нового набора инструментов.

Описание инфраструктуры

Разработанное решение состоит из нескольких компонентов, отношения между которыми изображены на рисунке 1. На рисунке цветами обозначены разработанные модули.

- Серверы с символьными машинами — служат в качестве механизма исследования графа исполнения;
- Коннектор — реализует протокол взаимодействия и логику управления символьным исполнением на серверах со стороны модуля валидации;

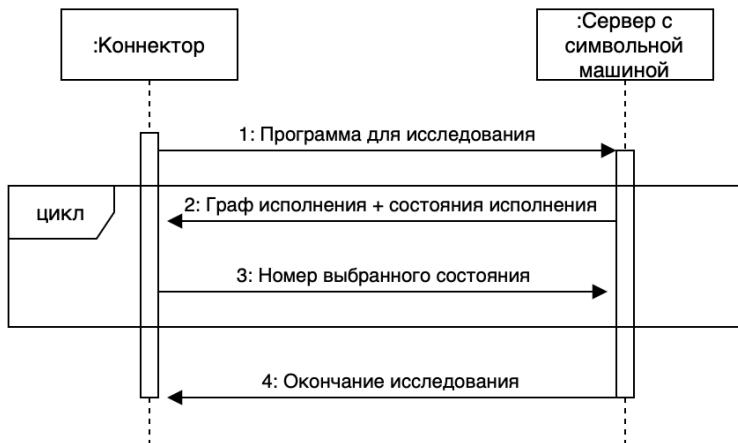


Рис. 2: Диаграмма последовательности протокола взаимодействия

- Брокер — создает серверы с символьными машинами и управляет соединениями с ними;
- Модуль обучения — осуществляет обучение нейронной сети;
- Модуль валидации — осуществляет валидацию, взаимодействует с сервером через Коннектор;
- ONNX конвертер — конвертирует обученные графовые нейронные сети (GNN) в формат ONNX [1].

Во время обучения нейронная сеть проходит через две фазы: тренировочную и валидационную. В тренировочной фазе сеть обучается выбирать оптимальный путь, а в валидационной фазе оценивается ее качество с помощью взаимодействия с сервером символьной машины. Так как символьная машина работает сравнительно медленно, валидационная фаза использует несколько серверов. Для создания и распределения серверов по задачам валидационной фазы был создан медиатор, обозначенный на рисунке как Брокер. Брокер создает сервера с символьными машинами и выдает их адреса Коннектору.

Универсальный протокол взаимодействия

Коннектор инкапсулирует логику взаимодействия с сервером. Он реализует протокол взаимодействия со стороны модуля валидации и отвечает за

обработку ошибок в логике взаимодействия. Коннектор может получать и обрабатывать сообщения, содержащие актуальное состояние графа исполнения и информацию о завершении работы символьной машины. Коннектор отправляет серверу сообщения о начале работы символьной машины и выбранном для исследования состоянии исполнения. Диаграмма последовательности процесса взаимодействия представлена на рисунке 2.

Обеспечение внедряемости результатов

Для внедряемости результатов обучения был разработан модуль экспорта графовых нейронных сетей в формат ONNX, предназначенный для представления моделей машинного обучения. ONNX определяет общий формат файлов, чтобы разработчики могли использовать модели в различных инструментах, средах выполнения и компиляторах [1].

Модуль сравнения результатов

Для получения данных и визуализации результатов экспериментов был разработан модуль сравнения результатов работы символьной машины с использованием различных стратегий выбора пути. Модуль запускает процесс символьной машины с запросом на генерацию тестов для выбранной программы и считывает результаты. После получения результатов производится сравнение стратегий с помощью визуализатора.

На рисунке 3 изображен график, созданный визуализатором. На нем представлено сравнение результатов работы символьной машины с использованием модели, обученной с помощью разработанной инфраструктуры, с существующими стратегиями.

Заключение

В результате работы была реализована инфраструктура для обучения графовых нейронных сетей выбору оптимального пути для символьного исполнения, позволяющая обучать нейронные сети во время взаимодействия с символьными машинами, внедрять обученные модели и сравнивать стратегии выбора пути.

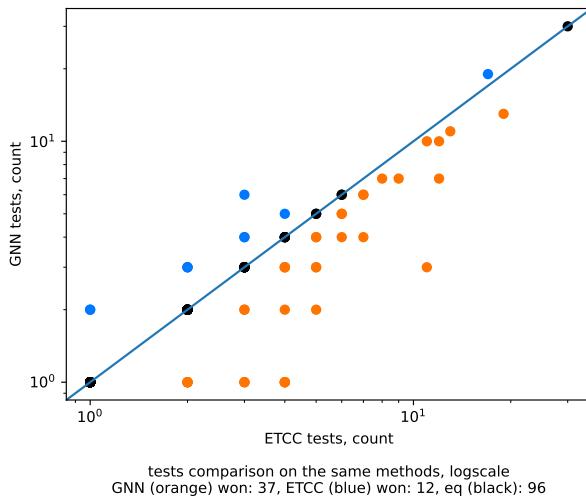


Рис. 3: Сравнение по количеству сгенерированных тестов с существующей стратегией на методах с одинаковым покрытием для двух стратегий. Оранжевым отмечены точки, где предложенный подход выигрывает сравнение.

Список литературы

- [1] Junjie Bai, Fang Lu, Ke Zhang и др. *ONNX: Open Neural Network Exchange*. <https://github.com/onnx/onnx>. Accessed: 2024-05-13. 2019.
- [2] J. Burnim и K. Sen. «Heuristics for Scalable Dynamic Test Generation». B: *Proceedings of the 23rd IEEE/ACM International Conference on Automated Software Engineering*. ASE '08. USA: IEEE Computer Society, 2008, c. 443–446. isbn: 9781424421879. doi: 10.1109/ASE.2008.69. url: <https://doi.org/10.1109/ASE.2008.69>.
- [3] Cristian Cadar, Daniel Dunbar и Dawson Engler. «KLEE: Unassisted and Automatic Generation of High-Coverage Tests for Complex Systems Programs». B: *Proceedings of the 8th USENIX Conference on Operating Systems Design and Implementation*. OSDI'08. San Diego, California: USENIX Association, 2008, c. 209–224.
- [4] Cristian Cadar и Koushik Sen. «Symbolic execution for software testing: three decades later». B: *Commun. ACM* 56.2 (2013), c. 82–90. issn: 0001-0782. doi: 10.1145/2408776.2408795. url: <https://doi.org/10.1145/2408776.2408795>.

- [5] Sooyoung Cha и др. «Enhancing Dynamic Symbolic Execution by Automatically Learning Search Heuristics». B: *IEEE Transactions on Software Engineering* 48.9 (2022), с. 3640—3663. doi: 10 . 1109 / TSE . 2021 . 3101870.
- [6] Keith D. Cooper и Linda Torczon. «Chapter 4 - Intermediate Representations». B: *Engineering a Compiler (Third Edition)*. Под пед. Keith D. Cooper и Linda Torczon. Third Edition. Philadelphia: Morgan Kaufmann, 2023, с. 159—207. isbn: 978-0-12-815412-0. doi: <https://doi.org/10.1016/B978-0-12-815412-0.00010-3>. url: <https://www.sciencedirect.com/science/article/pii/B9780128154120000103>.
- [7] Patrice Godefroid, Michael Y. Levin и David Molnar. «SAGE: Whitebox Fuzzing for Security Testing: SAGE Has Had a Remarkable Impact at Microsoft.» B: *Queue* 10.1 (2012), с. 20–27. issn: 1542-7730. doi: 10 . 1145 / 2090147 . 2094081. url: <https://doi.org/10.1145/2090147.2094081>.
- [8] Jingxuan He и др. «Learning to Explore Paths for Symbolic Execution». B: *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*. CCS '21. Virtual Event, Republic of Korea: Association for Computing Machinery, 2021, с. 2526—2540. isbn: 9781450384544. doi: 10.1145/3460120.3484813. url: <https://doi.org/10.1145/3460120.3484813>.
- [9] James C. King. «Symbolic Execution and Program Testing». B: *Commun. ACM* 19.7 (1976), с. 385—394. issn: 0001-0782. doi: 10 . 1145 / 360248 . 360252. url: <https://doi.org/10.1145/360248.360252>.
- [10] Franco Scarselli и др. «The Graph Neural Network Model». B: *IEEE Transactions on Neural Networks* 20.1 (2009), с. 61—80. doi: 10 . 1109 / TNN . 2008 . 2005605.
- [11] Symflower. *Symflower - Automated Unit Testing for Software Developers*. <https://symflower.com/en/>. Accessed: 2024-05-20.
- [12] Christopher J. C. H. Watkins и Peter Dayan. «Q-learning». B: *Machine Learning* 8.3 (май 1992), с. 279—292. issn: 1573-0565. doi: 10 . 1007 / BF00992698. url: <https://doi.org/10.1007/BF00992698>.
- [13] Jie Wu, Chengyu Zhang и Geguang Pu. «Reinforcement Learning Guided Symbolic Execution». B: *2020 IEEE 27th International Conference on Software Analysis, Evolution and Reengineering (SANER)*. 2020, с. 662—663. doi: 10 . 1109 / SANER48275 . 2020 . 9054815.

Синтез кода инициализации объектов для символического исполнения

Паршин М.А., СПбГУ, Санкт-Петербург mxprshn@gmail.com

Аннотация

Многие инструменты автоматической генерации тестов основаны на технике символьного исполнения. Символьное исполнение позволяет для каждого пути исполнения программы получить конкретные данные, на которых он достигается, и обеспечить полное тестовое покрытие кода.

В случае исследования объектно-ориентированных программ синтез кода тестов по состояниям символьного исполнения является нетривиальной задачей. Наивный подход к её решению порождает трудночитаемый код, использующий рефлексию.

В настоящей работе предложен алгоритм синтеза кода тестов по состояниям символьного исполнения, основанный на вычислении слабейших предусловий путей в методах. Прототип предложенного алгоритма реализован в символьной машине V#¹. В результате экспериментов для 43% состояний символьного исполнения были получены человекочитаемые тесты. Медианное время поиска последовательностей для метода составило около 5% от времени символьного исполнения.

Введение

Тестирование — неотъемлемый этап цикла разработки программного обеспечения, который часто подразумевает рутинную работу. Чтобы облегчить тестирование, разрабатываются инструменты для автоматической генерации тестов. Одна из технологий, лежащих в основе данных инструментов — символьное исполнение. Символьное исполнение [1] — техника статического анализа кода, заключающаяся в исполнении программы на *символьной памяти*, в которой данные представляются как термы формального языка. При этом для каждого пути исполнения выводится условие на термы, при выполнении которого он достигается — *условие пути*. С помощью SMT-решателя²[2] можно получить конкретные данные, удовлетворяющие усло-

¹<https://github.com/VSharp-team/VSharp>, дата обращения — 14.05.2024

²Satisfiability Modulo Theory, выполнимость в теориях

вию пути (*модель*), что позволит сгенерировать тест, покрывающий соответствующий путь.

Для объектно-ориентированных языков генерация тестов при помощи символьного исполнения затруднена тем, что объекты обязаны удовлетворять внутренним контрактам класса. Вследствие этого не по всем состояниям символьного исполнения можно получить тест. Например, если класс, представляющий односвязный список, содержит поле, отвечающее за его размер, оно никогда не принимает отрицательное значение при использовании публичных методов списка. При этом в результате символьного исполнения могут быть получены состояния с отрицательными значениями данного поля. Если же условие пути не противоречит внутренним контрактам класса, то возникает задача поиска *инициализирующей последовательности*. Инициализирующая последовательность — это последовательность вызовов публичных методов, в результате исполнения которой будут получены объекты, удовлетворяющие условию пути.

В данной работе предложен алгоритм поиска инициализирующих последовательностей. Основная идея предложенного алгоритма заключается в том, что инициализирующая последовательность строится от конца к началу. Производится перебор методов, позволяют построить объект данного типа. Чтобы определить, может ли вызов очередного метода быть присоединен к текущей последовательности, вычисляется *слабейшее предусловие* пути. В случае, если оно выполнимо, метод добавляется к последовательности.

Предложенный алгоритм был реализован на уровне прототипа в символьной виртуальной машине V#. В результате проведённых экспериментов для 43% состояний символьного исполнения были найдены инициализирующие последовательности. Медианное время, затраченное на их поиск для одного метода, составило около 5% от медианного времени его символьного исполнения.

Слабейшие предусловия

Пусть π — путь исполнения программы. *Постусловием пути* называется условие R на состояние программы, которое должно выполняться после исполнения инструкций из пути π . *Предусловием пути* $P(\pi, R)$ называется условие на состояние программы, из выполнимости которого следует, что после исполнения инструкций из пути π постусловие R будет выполняться. Соответственно, слабейшим предусловием пути $wp(\pi, R)$ называется такое предусловие пути, что любое другое следует из него.

В работе [8] предложен алгоритм вычисления слабейших предусловий

при помощи символьного исполнения.

Существующие решения

Расширение *Seeker* для символьной виртуальной машины *Pex* [5] решает задачу поиска инициализирующих последовательностей по заданному пути исполнения. «Скелеты»³ последовательностей исполняются символьно целиком, при этом отсутствует способ проверять их на корректность ещё на этапе их построения. Это приводит к расширению пространства поиска и, в результате, к увеличению времени, требуемого на исследование. С другой стороны, в работе *Seeker* также предлагается алгоритм статического анализа, позволяющий добавлять в «скелет» наиболее релевантные методы.

В работе [3] задача генерации инициализирующих последовательностей для символьного исполнения решается использованием инструментов эволюционного тестирования. Для этого условие пути конвертируется в функцию приспособленности для генетического алгоритма, что не применимо к сложным условиям пути, учитывающим, например, взаимодействие с «моковыми»⁴ объектами. Метод, предложенный в настоящей работе, может быть расширен для поддержки таких случаев.

В работах [4; 6; 7] также описан поиск инициализирующих последовательностей, но этот процесс рассматривается как самостоятельная технология тестирования, а не как вспомогательная задача для символьного исполнения.

Описание алгоритма

В данном разделе описан алгоритм, решающий задачу поиска инициализирующей последовательности по данному состоянию символьного исполнения. Состояние символьного исполнения представляет собой пару (π, μ) , где π — условие пути, а μ — символьная память.

Входными данными алгоритма является состояние (π_0, μ_0) символьного исполнения метода M , для которого требуется найти инициализирующую последовательность.

³англ. skeletons

⁴англ. mock

Состояние алгоритма

Состояние алгоритма — это тройка (sx, A, p) , где sx — суффикс последовательности, A — множество аргументов, а p — предусловие суффикса.

Суффикс последовательности представляет собой кортеж $(m_n, \dots, m_2, m_1, M)$, где m_1, m_2, \dots, m_n — методы последовательности (например, конструкторы), а M — тестируемый метод. Начальное значение — суффикс (M) , состоящий только из тестируемого метода.

Множество A содержит аргументы методов из суффикса, которые еще не были проинициализированы в нём. Изначально множество A содержит все аргументы метода M .

Предусловие p является логической формулой от аргументов из множества A . Все состояния исполнения программы, удовлетворяющие p , удовлетворяют π_0 после исполнения методов m_n, \dots, m_2, m_1 . Начальное значение — π_0 .

Композиция

Пусть $s_0 = ((m_n, \dots, m_2, m_1, M), A_0, p_0)$ — состояние алгоритма, а $\sigma = (\pi, \mu)$ — состояние символьного исполнения метода m_{n+1} . Определим *композицию* $s_1 = s_0 \circ \sigma$ (Рис. 1) как состояние алгоритма (sx_1, A_1, p_1) , полученное следующим образом:

- 1: $sx_1 \leftarrow (m_{n+1}, m_n, \dots, m_2, m_1, M)$
- 2: $A_1 \leftarrow A_0 \setminus A_{m_{n+1}}^- \cup A_{m_{n+1}}^+$
- 3: $p_1 \leftarrow wp(\pi, p_0)$

В первой строке к текущему суффиксу добавляется новый метод m_{n+1} . Во второй строке множество аргументов обновляется: из него вычитается множество $A_{m_{n+1}}^-$ — множество аргументов, которые были инициализированы вызовом m_{n+1} , и к нему добавляется множество $A_{m_{n+1}}^+$ аргументов метода m_{n+1} . В третьей строке за новое предусловие суффикса принимается слабейшее предусловие wp для текущего предусловия суффикса и условия пути π .

Процесс работы

Общая схема алгоритма представлена на рис. 2.

SymbolicMethodExplorer — символьная машина, осуществляющая символьное исполнение кода методов, из которых могут строиться последовательности. Получает на вход дескрипторы методов, которые необходимо ис-

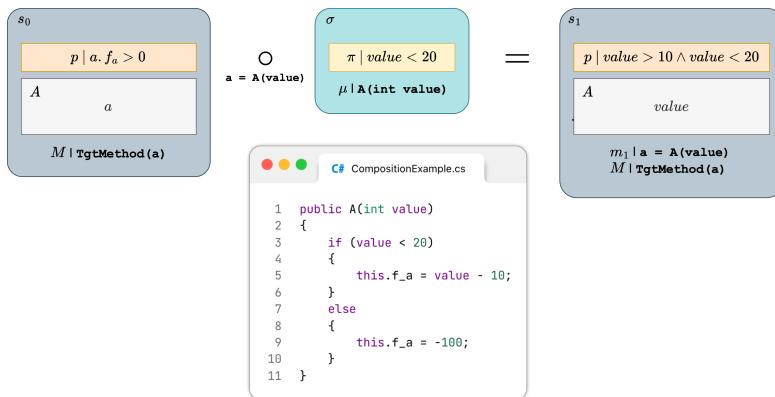


Рис. 1: Схема композиции

полнить, и в асинхронном режиме возвращает завершённые состояния символьного исполнения по мере исследования.

Queue — очередь, хранящая состояния алгоритма и соответствующие им состояния символьного исполнения, полученные от *SymbolicMethodExplorer*. Из очереди можно получить пару из состояния алгоритма и состояния символьного исполнения метода-кандидата на присоединение к последовательности.

Composer осуществляет композицию полученной из *Queue* пары состояний.

Пусть (sx_n, A_n, p_n) — результат композиции. Если p_n невыполнимо, то алгоритм продолжает работу. Если p_n выполнимо, и при этом $A_n = \emptyset$ или содержит только аргументы примитивных типов (например, *int*, *flow*, *char* и др.), то алгоритм завершает работу. В таком случае искомая последовательность найдена, аргументы примитивных типов могут быть найдены с помощью SMT-решателя. Если же это условие не выполняется, но p_n выполнимо, то s_n добавляется в очередь.

SequenceElementSelector осуществляет выбор методов-кандидатов на присоединение к текущему суффиксу последовательности. Для этого он производит анализ методов на то, какие поля ими используются.

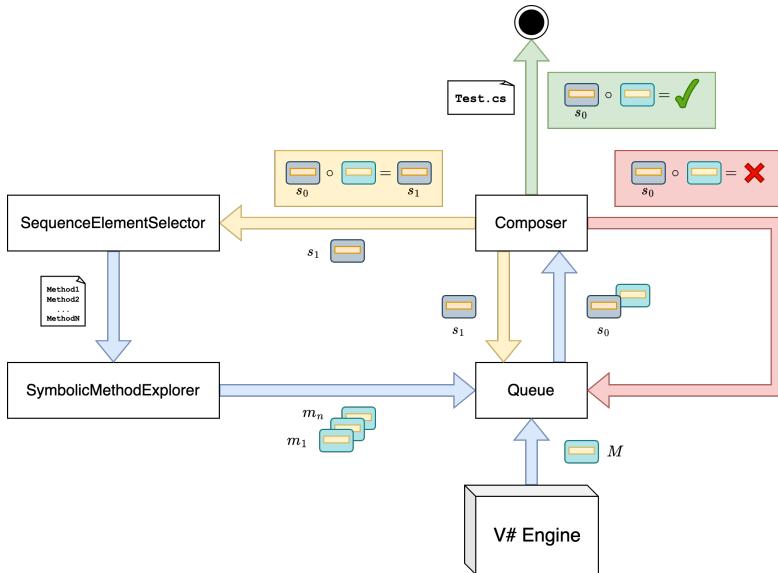


Рис. 2: Общая схема алгоритма

Реализация и эксперименты

Прототип предложенного алгоритма был реализован в символьной виртуальной машине V# на языке F#.

Эксперименты проводились на методах из проектов *btcpayserver*⁵ и *openra*⁶ (Табл. 1). На данный момент множество методов, из которых строится последовательность, ограничено конструкторами классов и методами, устанавливающими значения свойств⁷. В связи с этим, в тестовую выборку были включены только методы, последовательности для которых могли быть сгенерированы с учётом этих ограничений.

Машина, на которой проводились эксперименты — *MacBook Pro* с процессором *Apple M1 Pro* (8 ядер) и 16 ГБ RAM. ОС — *macOS 12.5.1*.

В начале для каждого метода из выборки запускалась символьная машина с лимитом времени 120 секунд на метод, затем для каждого из полученных состояний запускался алгоритм с лимитом времени 15 секунд на состояние. Медианное время для каждого из этапов приведено в табл. 1. Медианное вре-

⁵<https://github.com/btcpayserver/btcpayserver>, дата обращения — 14.05.2024

⁶<https://github.com/OpenRA/OpenRA>, дата обращения — 14.05.2024

⁷англ. *setters*, «сеттеры»

Количество методов	57
Количество состояний символьного исполнения	214
Медианное время исследования одного метода	19 сек
Медианное время генерации последовательностей для одного метода	1 сек

Таблица 1: Характеристики тестовой выборки и результаты экспериментов

мя поиска последовательностей для состояний символьного исполнения метода составило около 5% от медианного времени символьного исполнения.

На рис. 3 приведена гистограмма распределения длин сгенерированных последовательностей. 74% найденных последовательностей состоят из двух трёх методов. В 49% случаев был исчерпан лимит времени на поиск последовательностей. Это может быть связано с тем, что искомой последовательности не существует, или с ограниченным пространством поиска. Идентификация данных случаев требует дополнительной трудоёмкой работы, и запланирована на будущее. Наличие ошибочных запусков говорит о возникновении в процессе исследования случаев, не поддержанных в текущей реализации алгоритма или ядре символьной машины V#. Данные случаи включают в себя инициализацию «моковых» объектов символьной машиной, а также вызовы «внешних»⁸ методов в среде .NET в процессе символьного исполнения.

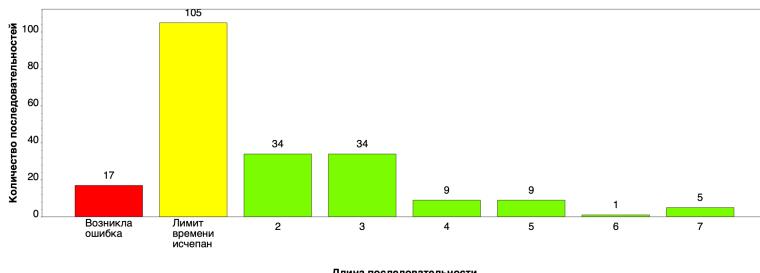


Рис. 3: Гистограмма распределения длин сгенерированных последовательностей

⁸англ. extern methods

Заключение

В представленной работе предложен алгоритм поиска инициализирующих последовательностей методов для состояний символьного исполнения, основанный на вычислении слабейших предусловий. Проведены эксперименты с прототипом алгоритма, реализованным в символьной машине V#. В 43% случаев были получены человекочитаемые тесты по состояниям символьного исполнения. Медианное время работы алгоритма составило около 5% от медианного времени символьного исполнения.

Список литературы

1. *Baldoni R., Coppa E., D'Elia D. C., Demetrescu C., Finocchi I.* A Survey of Symbolic Execution Techniques // ACM Comput. Surv. — New York, NY, USA, 2018. — Т. 51, № 3.
2. *Biere A., Heule M., Maaren H. van.* Handbook of Satisfiability. — IOS Press, 2009. — (Frontiers in Artificial Intelligence and Applications). — ISBN 9781607503767.
3. *Braione P., Denaro G., Mattavelli A., Pezzè M.* Combining symbolic execution and search-based testing for programs with complex heap inputs // Proceedings of the 26th ACM SIGSOFT International Symposium on Software Testing and Analysis. — Santa Barbara, CA, USA : Association for Computing Machinery, 2017. — С. 90—101. — (ISSTA 2017). — ISBN 9781450350761.
4. *Martena V., Orso A., Pezzè M.* Interclass testing of object oriented software //. — 02.2002. — С. 135—144. — ISBN 0-7695-1757-9.
5. *Thummalapenta S., Xie T., Tillmann N., Halleux J., Su Z.* Synthesizing Method Sequences for High-Coverage Testing //. Т. 46. — 10.2011. — С. 189—206.
6. *Xie T., Marinov D., Schulte W., Notkin D.* Symstra: A Framework for Generating Object-Oriented Unit Tests Using Symbolic Execution // Tools and Algorithms for the Construction and Analysis of Systems / под ред. N. Halbwachs, L. D. Zuck. — Berlin, Heidelberg : Springer Berlin Heidelberg, 2005. — С. 365—381. — ISBN 978-3-540-31980-1.
7. *Zhang Y., Zhu R., Xiong Y., Xie T.* Efficient Synthesis of Method Call Sequences for Test Generation and Bounded Verification // Proceedings of the 37th IEEE/ACM International Conference on Automated Software Engineering. — , Rochester, MI, USA, Association for Computing Machinery, 2023. — (ASE '22). — ISBN 9781450394758.

8. *Мисонижник А. В., Костюков Ю. О., Костицын М. П., Мордвинов Д. А., Козлов Д. В.* Генерация слабейших предусловий программ с динамической памятью в символьном исполнении // Научно-технический вестник информационных технологий, механики и оптики. — 2022. — Т. 22, № 5. — С. 982—991.

Оптимизация энергопотребления Android-приложения на основе применения «зелёных» паттернов

Тарбеев А.В., СПбГУ, Санкт-Петербург andrey@tarbeev.net,
Сабашный В.Е., Генеральный директор ООО «ЛАНИТ-ТЕРКОМ», Санкт-Петербург
vadim.sabashny@lanit-tercom.com,
Сартасов С.Ю., СПбГУ, Санкт-Петербург stanislav.sartasov@spbu.ru

Аннотация

Энергоэффективность Android-приложений может достигаться путем использования специализированных оптимизаций со стороны разработчиков. В данной работе рассматривается вариант энергоэффективной оптимизации с использованием коллекции HashedMap из библиотеки Apache Commons Collections и переработкой паттерна декоратор.

Введение

В наше время мобильные устройства пользуются большой популярностью, и их число с каждым днем растет. С ростом популярности мобильных устройств выросли и требования к энергоэффективности аппаратных средств и программного обеспечения.

Важным пунктом на пути к выгодному использованию энергии является участие в процессе разработки всех, кто хотя бы как-то связан с мобильными устройствами. Инженеры создают энергоэффективные чипы и процессоры, а разработчики начинают пользоваться программными средствами для анализа потребления энергии приложением. Именно благодаря общим усилиям можно добиться максимальной энергоэффективности. Если, например, приложение будет написано неэффективно, то каким энергоэффективным не был бы процессор, устройство не сможет использовать запас заряда оптимально.

На помощь разработчикам программного обеспечения приходят не только инструменты профилирования, но и подходы в написании кода, получившие название «зелёных» паттернов, которые, как выясняется, тоже вносят свой вклад в энергопотребление. В последнее время ведутся работы по анализу влияния паттернов проектирования на потребление энергии.

В одной из работ [1] среди разработчиков был проведен опрос, чтобы выяснить их осведомленность о влиянии использования паттернов на энергопотребление. Результаты говорят о том, что большинство не учитывает их возможное влияние на энергозатраты при разработке. Исходя из результатов,

предположение о том, что open-source проекты содержат неэнергоэффективный код, является справедливым.

В данной работе предлагается провести обзор «зелёных» паттернов и далее внедрить набор оптимизационных решений, которые будут основано на использовании энергоэффективных структурах данных и на переработке паттерна декоратор, и оценить энергосберегающий эффект.

«Зелёные» паттерны

Понятие «зелёных» паттернов подразумевает под собой набор подходов, которые позволяют снизить энергопотребление приложения. Например, преимущественное использование темных оттенков для UI элементов, использование кэширования, выбор WiFi вместо мобильной сети. Их списки представлены в статьях [2, 3].

В статье [4] включено описание «зелёных» подходов, описания их влияния на энергопотребление. Например, вместо автоматического выполнения действия, являющегося энергозатратным, рекомендуется выполнять их по требованию пользователя. Или, например, заменить использование экрана в качестве источника информации на что-либо другое, например, на звуковые уведомления.

В работе [5] авторы сравнивали не одиночное применение «зелёных» подходов, а их совокупность. В результате они пришли к выводу, что некоторые комбинации могут негативно влиять на энергопотребление по сравнению с их отдельным применением.

В другой работе [6] авторы провели анализ влияния коллекций Java на энергопотребление. Они установили, что использование некоторых коллекций, например, *HashMap*, ощутимо влияет на использование энергии. К тому же было установлено, что при смене коллекций на более энергоэффективные аналоги потребление снизилось сильнее у настольных систем в сравнении с мобильными системами. Дополнительно, был реализован инструмент, позволяющий давать рекомендации по замене коллекций.

В одной из работ [1] была затронута тема возможного влияния параметров, спрятанных внутри кода, на энергопотребление Android-приложений. Примеры таких параметров: размеры буферов, размеры кэшей, количество потоков, частоты опросов и даже расположения UI элементов. Был реализован инструмент, который автоматически находил такие параметры, затем изменял их в надежде выяснить изменения в энергопотреблении. После каждого изменения авторы доступными средствами операционной системы снимали показания нагруженности аппаратных модулей, в последствии эти данные

использовались в энергетической модели, где и происходил расчет энергопотребления конкретного модуля. В результате проведенной работы авторы статьи пришли к выводу, что параметры практически не влияют на энергопотребление.

По результатам проведенного обзора «зелёных» паттернов выяснилось, что их использование в проектах с учетом их малой популярности среди разработчиков является набирающим популярность трендом.

Паттерны проектирования

Одной из значимых работ в области анализа влияния применения паттернов проектирования на энергопотребление является статья [7] C. Sahin et al. В ней авторы показали, что использование ряда паттернов проектирования может оказываться на энергопотреблении как в сторону увеличения, так и в сторону уменьшения. Так, по словам авторов, в *C++* паттерн *Decorator* увеличил энергопотребление на $\approx 700\%$, а паттерн *Flyweight* уменьшил на $\approx 58\%$. В дополнение, паттерны, находящиеся по смыслу в одной категории (*creational, structural, behavioral*), не зависят от своей категории и могут как увеличивать, так и уменьшать энергопотребление.

В одной из работ [8] авторы провели анализ *Decorator* паттерна в *Java*. В нем сравнивалось энергопотребление книжного примера кода кофейного автомата с паттерном и без него. Версия с рефакторингом, то есть без *Decorator* паттерна, использовала на $\approx 96\%$ меньше энергии.

В работе [9] авторы оценили влияние применения ООП принципов, таких как наследование, полиморфизм, перегрузка операторов, на энергоэффективность. Наследование и полиморфизм не несут значительного влияния на энергопотребление и производительность, перегрузка операторов в свою очередь оказывает существенное влияние на энергоэффективность и производительность. Это связано с тем, что использование перегрузки может приводить к созданию временных объектов для промежуточных результатов, что и приводит к снижению производительности и увеличению энергопотребления.

Несмотря на то, что паттерны проектирования вносят структурированность и понятность в программную составляющую продукта, использование некоторых паттернов значительно влияет на энергозатраты, актуальной задачей остается их преобразование в более энергоэффективные аналоги.

По результатам проведенного обзора выяснилось, что внедрение «зелёных» паттернов и энергоэффективных паттернов проектирования является важным этапом в повышении энергоэффективности приложений, поэто-

му полученные результаты будут использоваться в процессе внедрения этих практик в open-source проект с дальнейшей оценкой энергопотребления.

Оптимизация

Описание экспериментов

В качестве тестовых Android-приложений были выбраны Remixed Dungeon[11] и Hentoid[12]. Они доступны публично, активно используются и развиваются.

Для проведения замеров были использованы смартфон (Vertex Luck L130, Cortex-A53, 2GB RAM), мультиметр (UNI-T UT803) и лабораторный генератор (QJE QJ3003H).

Смартфон, мультиметр и генератор были соединены в последовательную цепь. На генераторе фиксировалось напряжение, имеющее значение, которое совпадает со значением, указанным на аккумуляторе смартфона. Для минимизации влияния внешних факторов на замеры на смартфоне была выставлена минимальная яркость, отключены Wi-Fi и Bluetooth. Для фиксации тестового сценария использования приложений были написаны скрипты с использованием команд ADB (Android Debug Bridge). Время одного прогона составляло 15 минут. Также для устранения случайной ошибки замеров количество прогонов составляло десять повторений.

Использование энергоэффективных структур данных

Помимо стандартной библиотеки коллекций Java существуют сторонние библиотеки, например, Apache Commons Collections[10], которая содержит более оптимизированные в плане энергопотребления реализации коллекций. Нас будет интересовать сторонняя коллекция HashedMap из Apache Commons Collections. Она будет использоваться вместо стандартной реализации HashMap.

В Remixed Dungeon и Hentoid обычная Java-коллекция HashMap была заменена на HashedMap коллекцию из Apache Commons библиотеки. По результатам замеров для двух приложений были получены результаты 0.5% и 0.6% соответственно.

Переработка паттерна декоратор

Существует несколько вариантов оптимизации паттерна декоратор. Создание объекта — это всегда ресурсоемкий процесс, заключающийся в поиске и аллокации необходимого пространства для новых объектов. Также из-за большого числа небольших создаваемых объектов растет нагрузка на сборщик мусора. В работе же будет рассмотрен вариант оптимизации с объединением всех декорируемых объектов для значительного сокращения числа создаваемых объектов.

В Remixed Dungeon приложении таким образом было сокращено число инстанцируемых сложных промежуточных объектов. В результате прирост составил 4.1%.

Вывод

Оптимизации понесли за собой ряд других проблем, которые необходимо принимать во внимание при решении об их внедрении.

Во-первых, это увеличение размера собранного приложения при использовании сторонней библиотеки для коллекций, Remixed Dungeon стало весить больше на 5 мегабайт, что не является критичным показателем в наше время. Однако для устройств, которые ограничены в количестве доступной памяти, это может стать важным фактором в решении о внедрении данной оптимизации.

Во-вторых, паттерны проектирования в первую очередь служат для того, чтобы написанный код был понятным и легкорасширяемым. Однако, стараясь угнаться за энергоэффективностью, мы теряем эти параметры. Стоит отметить, что известной практикой в игровой индустрии является использование анти-паттернов для повышения производительности в случае, если остальные способы оптимизации уже были внедрены, сюда же можно добавить и энергопотребление.

Заключение

Экспериментальным путем было показано, что влиять на энергопотребление Android-приложения возможно со стороны кода. Для приложений Remixed Dungeon и Hentoid после внедрения энергоэффективной коллекции HashedMap выигрыш в энергопотреблении составил 0.5% и 0.6% соответственно. Для приложения Remixed Dungeon после переработки паттерна декоратор выигрыш в энергопотреблении составил 4.1%.

Список литературы

- [1] Xu, Qiang and Davis, James C. and Hu, Y. Charlie and Jindal, Abhilash. An Empirical Study on the Impact of Deep Parameters on Mobile App Energy Usage // 2022 IEEE International Conference on Software Analysis, Evolution and Reengineering (SANER). — С. 844–855. <http://dx.doi.org/10.1109/SANER53432.2022.00103>
- [2] Nelson Gregório, João Paulo Fernandes, João Bispo, Sérgio Medeiros. E-APK: Energy Pattern Detection in Decompiled Android Applications // Proceedings of the XXVI Brazilian Symposium on Programming Languages. — С. 50–58. <http://dx.doi.org/10.1145/3561320.3561328>
- [3] Cruz Luis, Abreu Rui. Catalog of Energy Patterns for Mobile Applications // Empirical Softw. Engg. — С. 2209–2235. <https://doi.org/10.1007/s10664-019-09682-0>
- [4] Daniel Feitosa and Luis Cruz and Rui Abreu and João Paulo Fernandes and Marco Couto and João de Sousa Saraiva. Patterns and Energy Consumption: Design, Implementation, Studies, and Stories // Software Sustainability. <https://api.semanticscholar.org/CorpusID:240707094>
- [5] Couto, Marco and Saraiva, João and Fernandes, João Paulo. Energy Refactorings for Android in the Large and in the Wild // 2020 IEEE 27th International Conference on Software Analysis, Evolution and Reengineering (SANER). — С. 217–228. <http://dx.doi.org/10.1109/SANER48275.2020.9054858>
- [6] Oliveira, Wellington and Oliveira, Renato and Castor, Fernando and Pinto, Gustavo and Fernandes, João Paulo. Improving Energy-Efficiency by Recommending Java Collections // Empirical Softw. Engg. — С. 3–45. <https://doi.org/10.1007/s10664-021-09950-y>
- [7] Sahin, Cagri and Cayci, Furkan and Gutiérrez, Irene Lizeth Manotas and Clause, James and Kiamilev, Fouad and Pollock, Lori and Winbladh, Kristina. Initial explorations on design pattern energy usage // 2012 First International Workshop on Green and Sustainable Software (GREENS). — С. 55–61. <http://dx.doi.org/10.1109/GREENS.2012.6224257>
- [8] Connolly Bree, Déaglán and Cinnéide, Mel Ó. Removing Decorator to Improve Energy Efficiency // 2022 IEEE International Conference on Software Analysis, Evolution and Reengineering (SANER). — С. 902–912. <http://dx.doi.org/10.1109/SANER53432.2022.00108>

- [9] Maleki, Sepideh and Fu, Cuijiao and Banotra, Arun and Zong, Ziliang. Understanding the impact of object oriented programming and design patterns on energy efficiency // 2017 Eighth International Green and Sustainable Computing Conference (IGCC). — С. 1–6. <http://dx.doi.org/10.1109/IGCC.2017.8323605>
- [10] Apache Commons Collections. <https://commons.apache.org/proper/commons-collections/>
- [11] Remixed Dungeon <https://github.com/NYRDS/remixed-dungeon>
- [12] Hentoid <https://github.com/avluis/Hentoid>

Содержание

Вероятностные графические модели, нечеткие системы, мягкие вычисления и соиокомпьютинг, машинное обучение	3
Алимов П. Г., Гориховский В. И. Разработка пакета нейросетевой аппроксимации дифференциальных уравнений DEGANN	5
Владимирова Э. В. Целочисленное квантование для вывода при глубоком обучении	12
Егоров П. А. Внедрение оптимизационных алгоритмов в интерфейс фреймворка Streamlit	19
Ельцов Д. А. Дистилляция диффузионных моделей для создания 3D контента	26
Стельмах Т. Д. Детектирование искусственно сгенерированного научного текста	33
Вычислительная стохастика и статистические модели, многокритериальная стохастическая оптимизация	39
Алексеева Н. П., Подлеснов Я. С. Бустинговый алгоритм дисперсионного анализа на блок-схемах с медико-биологическими приложениями	41
Шаповал Е. А., Голяндина Н. Э. Подход к сравнению методов обнаружения разладки	48
Олейник М. В., Алексеева Н. П. О редукции параметров в моделях сложных распределений с применением в радиобиологии	56
Вычислительные задачи механики и астрономии	63
Русаков А. С., Аболмасов П. К. Сферическая адаптация MUSCL схемы для решения задачи растекающихся слоёв нейтронных звёзд	65
Зернов С. Н. Создание программного продукта для моделирования химической релаксации газа за ударной волной	72
Модели, методы и приложения тропической математики	75
Филатова А. А., Кривулин Н. К. Оценка альтернатив на основе парных сравнений в задаче об определении лидера группы	77
Параллельные алгоритмы, вэйвлетная обработка числовых потоков	85
Лившиц Л. П. Квазилинейная интерполяция минимальными сплайнами	87
Битепаж В. О. О приближении кусочно-постоянными финитными функциями и их обобщениями	89

Евдокимова Т. О., Иванцова О. Н. О курсе «Методика преподавания компьютерных наук» в непедагогических вузах	90
Макаров А. А., Савельева М. Ю., Шабунин А. Н. О распараллеливании одного метода бинаризации	97
Косогоров О. М., Макаров А. А., Макарова С. В. Вейвлетные схемы в обработке радиолокационных данных	99
Прикладная кибернетика и искусственный интеллект 101	
Арсеньев Д. Г., Кузнецов Н. В., Лобачев М. Ю. Анализ разрывной модели фазовой автоподстройки с пилообразной характеристикой фазового детектора	103
Антропова Е. Г. Нейронная сеть Хопфилда для решения задач оптимального управления	105
Кисиев Т. А., Мокаев Т. Н. Глобальная устойчивость нейронных сетей с недифференцируемыми активационными функциями	111
Королев А. С., Благов М. В. Автоматический контроль качества конвейеров данных	118
Латанов К. В., Благов М. В., Кузнецов Н. В. Telegram-бот для обработки стеганографии в PNG-файлах	125
Мажара Е. Н. Анализ инструментов быстрого захвата изменений данных	130
Мазяр В. А., Мокаев Т. Н. Изучение динамики клеточных автоматов с помощью нейронных сетей	136
Плотникова М. А. Пространственный анализ данных для исследования потенциала территории	143
Сафин А. Ф., Благов М. В. Разработка отказоустойчивого конвейера для доставки данных в режиме квазиреального времени	151
Распараллеливание в OpenMP и сплайновые аппроксимации 159	
Yang Yu., Yakubovich Y. Stochastic properties of a random Young diagram	161
Козгунов Н., Халаши М. Архитектура динамической децентрализованной большой языковой модели с использование технологии консенсусов	163
Системное программирование и программная инженерия 171	
Житихина М. А. Построение модели полосы с учетом контекста дороги	173
Казанцев А. А., Гориховский В. И. Эффективное оценивание параметров смеси распределений	178

Муравьев И. В. Оптимизация алгоритма контекстно-свободной до- стижимости, основанного на операциях линейной алгебры	195
Нигматулин М. В., Чистякова А. А., Григорьев С. В. Разработка инфраструктуры для обучения искусственных нейронных сетей вы- бору оптимального пути для символьного исполнения	202
Паршин М. А. Синтез кода инициализации объектов для символь- ного исполнения	209
Тарбеев А. В., Сабашный В. Е., Сартасов С. Ю. Оптимизация энер- гопотребления Android-приложения на основе применения «зелё- ных» паттернов	218

Организаторы конференции «Мат-мех. Наука 2024» благодарят спонсоров и партнёров конференции за неоценимую ресурсную и организационную помощь в проведении конференции и при подготовке сборника материалов.

В 2024 году нас поддержали:



Санкт-Петербургский
государственный
университет



Группа компаний
«ЯДРО»

Научное издание

Материалы весенней научно-практической конференции
по вопросам информатики, математики,
механики и астрономии
«Мат-мех. Наука 2024»

*29 апреля – 3 мая 2024 г.
Санкт-Петербург*

Компьютерная верстка: Д.В. Луцив

Издательство «ВВМ»
198095, Санкт-Петербург,
ул. Швецова,
д. 41, лит. А, пом. 06Н, Ч.П. 49, 50
E-mail: vvmpub@yandex.ru

Подписано в печать 01.07.2024. Формат 60 × 84 1/16.
Бумага офсетная. Гарнитура Times. Печать цифровая.
Усл. печ. л. 13,37. Тираж 100 экз. Заказ №2333.

Отпечатано в Издательстве ВВМ.
198095, Санкт-Петербург, ул. Швецова, 41.