ГОЛЕМ: Гибкий эволюционный дизайн графических представлений физических и цифровых объектов

Майя Пинчук, Григорий Киргизов, Любовь Явщикова, Николай Никитин, Ирина Деева, Каринэ Шакхян, Иван Борисов, Кирилл Жарков, Анна Калюжная

Университет ИТМО, г. Санкт-Петербург, Россия

Введение
Мы представляем GOLEM, оптимизационную платформу с открытым исходным кодом для автоматизированного проектирования графовых структур в различных научных областях. Она решает проблему поиска оптимальной топологии и параметров графа с помощью эволюционных алгоритмов и делает это модульным, независимым от предметной области способом. В статье описывается фреймворк и его гибкий подход к адаптации к предметной области. Экспериментальные исследования приводят несколько примеров использования GOLEM в различных областях: байесовских сетях, разработке лекарств и робототехнике.

КОНЦЕПЦИИ CCS
• Вычислительные методики; • Программное обеспечение и его разработка;

Ключевые слова
Эволюционный дизайн, Графический поиск, Открытый исходный код, Робототехника, Разработка лекарств

Справочный формат ACM:
Майя Пинчук, Григорий Киргизов, Любовь Ямщикова, Николай Никитин, Ирина Деева, Карине Шахкян, Иван Борисов, Кирилл Жирков и Анна Калюжная. 2024 год. GOLEM: Гибкий эволюционный дизайн графических представлений физических и цифровых объектов. В журнале Genetic and Evolutionary

Конференция по вычислениям (GECCO ’24 Companion), 14-18 июля 2024 г., Мелборн, Виктория, Австралия. ACM, Нью-Йорк, Нью-Йорк, США, 8 страниц. https://doi.org/10.1145/3638530.3664141

1 Вступление

Проблема поиска оптимальных структур в виде графов возникает во многих научных и промышленных областях [35]. Примеры варьируются от разработки вероятностных моделей, таких как байесовское сетевое моделирование (BN) [24], поиск по нейронной архитектуре (NAS) [9] и автоматическое проектирование конвейеров машинного обучения (AutoML) [15], до реальных приложений, таких как проектирование инженерных сооружений [40], робототехника. скелеты [31] и молекулярные структуры при разработке лекарственных препаратов [30]. Такие задачи требуют глубоких знаний в предметной области и проведения множества этапов прототипирования и

экспериментов, на что могут потребоваться значительные ресурсы. Таким образом, необходим инструмент, способный автоматизировать задачи графического дизайна.

Мы описываем этот класс задач как поиск и оптимизацию графов [6]. Он включает в себя нахождение оптимальной топологии (поиск) и параметров узлов (оптимизация). Характеристики этой задачи включают в себя: дискретное пространство задач, небольшой или средний размер графа (от 10 до 100 узлов), иногда с очень разнородными узлами, требующими множества различных параметров (например, AutoML, где каждый узел является ML-моделью со своим собственным набором гиперпараметров), и часто отсутствуют наборы данных для решения.

Этот курс является частью более широкой области изучения графов, которая включает в себя такие задачи, как классификация графов, узлов и связей, а также прогнозирование. Хотя в последние годы тема обучения на графах привлекла много внимания, особенно в связи с глубоким обучением и нейронными сетями на графах (GNN), и достигла впечатляющего прогресса во многих приложениях, например, в разработке лекарств [21], методы GNN часто неприменимы непосредственно к таким задачам. В частности, в области автоматизации по-прежнему доминируют метаэвристические методы [15], а методы глубокой нейронной сети показывают противоречивые результаты [16].

Метаэвристические методы [1], такие как эволюционный поиск или оптимизация роя частиц, представляют собой один из подходов к решению задач изучения графов. Эти методы не требуют градиентов и могут работать непосредственно с дискретными проблемными пространствами. Многолетний опыт использования этих методов показывает, что они могут достигать хороших результатов в сложных пространствах поиска [17], демонстрируя сопоставимые или даже лучшие результаты, чем методы нейронных сетей на основе градиента [27, 37]. Все вышесказанное делает его отличным решением для оптимизации графов, где пространство решений имеет дискретную природу.

Насколько нам известно, не существует доступного и универсального инструмента для поиска и оптимизации графиков, несмотря на общую формулировку задачи. Но почему такого инструмента нет? С одной стороны, большинство эффективных методов, предложенных в литературе, зависят от конкретной задачи или предметной области и их трудно обобщить [23, 46], с другой стороны, существующие универсальные системы оптимизации, как основанные на градиенте, так и метаэвристические [2], не могут быть применены к графам без большого набора известных решений, которые редко доступны в конкретных областях, или обширной работы по интеграции с метаэвристическими фреймворками, которые не предназначены для графиков.

Наш вклад направлен на то, чтобы восполнить этот пробел. Мы представляем GOLEM - независимую от предметной области платформу для оптимизации графов с помощью эволюционных методов. GOLEM изначально поддерживает дискретную оптимизацию графов, реализует специфичные для графов мутации и поддерживает встроенную структуру, что значительно снижает затраты на ее использование в новой области. GOLEM - это проект с открытым исходным кодом, доступный на GitHub под лицензией BSD-3. В статье мы описываем успешные применения GOLEM в

нескольких областях: вероятностное моделирование, разработка лекарств, робототехника, и сравниваем их с исходными данными.

2 РАБОТЫ ПО ТЕМЕ

Мы разделили существующие подходы к изучению графов на три группы: универсальные фреймворки оптимизации, методы, основанные на нейронной сети графов (GNN), и приложения для предметной области.

2.1 Универсальные фреймворки для оптимизации графов

В этой области существует несколько метаэвристических фреймворков - от исследовательских разработок до библиотек, готовых к производству. DEAP - популярный фреймворк для быстрого создания прототипов эволюционных алгоритмов. Два других популярных примера - Pymoo [2], библиотека для многоцелевой оптимизации с реальными значениями, и PyPop7 [8], которая представляет собой «чисто питоновскую библиотеку POPulation-based OPtimization для одноцелевых, реально-параметрических, „черных“ задач». Они включают разнообразный набор методов оптимизации «черного ящика», в том числе эволюционные алгоритмы.

Другая категория - градиентные эволюционные фреймворки, такие как EvoTorch или EvoJAX, основанные на фреймворках PyTorch и JAX соответственно. Хотя их основное применение - нейроэволюция, они могут быть использованы для оптимизации в любой области с реальными значениями и целью.

Существующие метаэвристические фреймворки нуждаются в мостике в область оптимизации графов. Например, теоретически DEAP может быть использован для оптимизации любой дискретной структуры. Однако это требует значительных усилий по реализации, поскольку DEAP предоставляет только базовую инфраструктуру для эволюции. Для оптимизации с помощью этих фреймворков можно тривиально преобразовать граф в его матрицу смежности, но эта задача усложняется для параметризованных графов или графов с нефиксированным размером.

2.2 Графические нейронные сети

Graph Neural Network (GNN)-графовая нейронная сеть, может рассматриваться как эффективный инструмент для проектирования графов. Методы на основе ГНС реализованы в таких библиотеках, как Jraph [11], PyG [10], DGL [45] и других. Они показывают современные результаты на большом количестве задач, связанных с графами (например, оптимальный поток энергии [34], поиск лекарств [20], NAS [33]). Однако основными задачами GNN являются предсказания графов, узлов и ребер. Существует несколько методов на основе GNN, направленных именно на поиск или генерацию графов: как специфических (например, поиск лекарств [28]), так и не связанных с конкретной областью (например, подходы для маркированных графов [12], решения на основе иерархических RNN [49], алгоритма случайных блужданий, генеративных моделей на основе баллов [32] и многомасштабных [51]).

Однако ГНН требуют больших наборов данных с известными решениями, которые довольно редко доступны в потенциальных областях применения. Даже если подходящие данные существуют, они могут вносить значительную погрешность [27].

Метаэвристические методы, с другой стороны, не требуют наборов данных для обучения и нуждаются только в объективной функции или, в крайнем случае, в нескольких образцовых решениях для сравнения. Методы GNN также менее применимы для небольших графов (от 10 до 100 узлов) и сильно неоднородных графов, в основном из-за недостатка данных в таких случаях для обобщения вкраплений узлов и графов.

2.3 Оптимизация графиков в определенных областях

Существует множество приложений в различных областях, направленных на оптимизацию графов. Мы выделяем только те из них, которые имеют отношение к рассматриваемым в статье примерам.

Одним из примеров является G2o: общий фреймворк для оптимизации графов в области робототехники и одновременной локализации и картографирования (SLAM). Его основная идея основана на наблюдении, что многие популярные задачи в робототехнике и компьютерном зрении, включая различные типы SLAM или настройку связки (BA), могут быть сформулированы как оптимизация функции ошибки по наименьшим квадратам, которая может быть представлена в виде графа [13].

Оптимизация графа охватывает структурное обучение БН, особенно когда набор данных принимает форму графа, где узлы соответствуют признакам, а ребра - зависимостям между ними. Граф служит моделью многомерного распределения, которое характеризует данные. Классические алгоритмы обучения BN используют тесты на условную независимость (CI) [19]. В этом направлении наиболее современным решением является PC-алгоритм [5]. Ряд алгоритмов обучения основан на формулировке задачи как оптимизационной задачи в пространстве возможных структур с оценочной функцией. К таким алгоритмам относятся алгоритм K2, Hill-Climbing, Sparse-Candidate. Существуют также гибридные подходы, сочетающие алгоритмы CI и алгоритмы, основанные на оценке [43]. Эти алгоритмы специфичны для обучения БН и не могут быть обобщены на различные сценарии, такие как узлы, содержащие наблюдения вместо признаков, или критерии оптимизации, отличные от мер качества для многомерных распределений. Наконец, многообещающие результаты дает привлечение непрерывной оптимизации к обучению БН [50].

Генерация молекул, ориентированная на достижение цели, может рассматриваться как задача оптимизации структуры графа [27]. Хотя представление строк с помощью SMILES [47] также довольно популярно [39], некоторые спорят об эффективности этого подхода, поскольку изменение символов часто приводит к недействительным структурам [18]. Существующие решения можно разделить на алгоритмы, использующие последние достижения в области глубокого обучения [44], и классические методы оптимизации, например эволюционные алгоритмы. Однако решения, основанные на машинном обучении, в значительной степени зависят от обучающих данных, поэтому такие модели могут быть предвзяты к известным молекулярным структурам, что негативно сказывается на их исследовательских возможностях.

3 ПОСТАНОВКА ЗАДАЧИ

В литературе существуют различные формулировки задач обучения графов [22, 35]. Задача, решаемая GOLEM, заключается в минимизации цели F в дискретном пространстве параметризованных направленных графов G с учетом набора произвольных графовых ограничений C. Задача заключается в нахождении оптимального графа G* = ⟨V , E, PV ⟩ с набором вершинных параметров PV.

$$G^* = \underset{\mathbb{G}}{argmin}\, F \; where \; \mathbb{G} = \{G \mid \mathbb{C}(G) \; is \; True\}$$

В многоцелевом случае минимизация позволяет найти Парето-фронт оптимальных решений. Любая доменная задача может быть сведена к этой оптимизационной задаче, если мы можем определить отображение A (где A означает Adapter) из пространства доменных структур S в параметризованные направленные графы вместе с обратным отображением R (где R означает Restore). Его роль будет раскрыта далее.

$$A : \mathbb{S} \rightarrow \mathbb{G} \; and \; R : \mathbb{G} \rightarrow \mathbb{S}$$

4 ПРЕДЛОЖЕННЫЙ ПОДХОД

В этом разделе мы опишем реализованный подход к заявленной задаче. Фреймворк GOLEM состоит из трех логических уровней:

• Ядро оптимизации, которое отвечает за работу с графами, алгоритмами оптимизации, целями, условиями остановки и т.д. Оно не зависит от предметной области и работает на внутреннем графическом представлении.

• Уровень спецификации предметной области, который отвечает за двунаправленное преобразование структур предметной области и внутренних графов. Это мост между конкретной предметной областью и универсальным ядром оптимизации.

• Инфраструктурный уровень, отвечающий за неосновные функции, такие как сериализация и визуализация, и технические детали оценки, такие как кэширование и распараллеливание. Это обеспечивает удобство использования, эффективность, интерпретируемость и воспроизводимость экспериментов.

4.1 Ядро оптимизации

Оптимизационное ядро - это набор абстракций, связанных с: графами, графовыми оптимизаторами, объектными функциями, адаптивными условиями остановки и конкретными реализациями алгоритмов оптимизации. Целью данного модуля является решение задачи, определенной в разделе 2.3.

Основной алгоритм оптимизации представляет собой реализацию многоцелевого эволюционного алгоритма на основе селекции SPEA-2 с операторами мутации и кроссовера, предназначенного для графов. Выбор известного SPEA-2 обусловлен его успешным применением в различных приложениях, связанных с графами [40]. По умолчанию используется селекция SPEA-2, однако есть возможность

использовать собственный алгоритм селекции или выбрать один из реализованных в GOLEM. Кроме того, мы добавили шаг верификации для учета специфических ограничений на графы.

Также часто бывает, что цель является частичной функцией, которую нельзя оценить для некоторых особей, например, задачи на основе моделирования с тайм-аутом. Если существует слишком много особей, которые не могут быть оценены, производительность эволюционного алгоритма может ухудшиться из-за уменьшения и непредсказуемости размера популяции. Чтобы решить эту практическую проблему, мы добавляем в алгоритм контроллер размножения, который реализует пропорциональный регулятор размера популяции для компенсации недействительных оценок.

На рисунке 1 представлена схема ядра оптимизации с интегрированным эволюционным алгоритмом. Входные параметры алгоритма оптимизации включают пространство поиска графов (доступные типы узлов и параметры узлов), целевую функцию и начальную популяцию графов. Также опционально могут быть введены ограничения и пользовательские операторы. При использовании эволюционных операторов (мутации, кроссинговеры) применяется отбор и правила ограничений графа, которые учитываются на каждой итерации на популяции индивидуальных решений. Контроллер размножения поддерживает стабильный размер популяции. После эволюции гиперпараметры лучших графов (параметры узлов графа) оптимизируются с помощью внешнего тюнера.

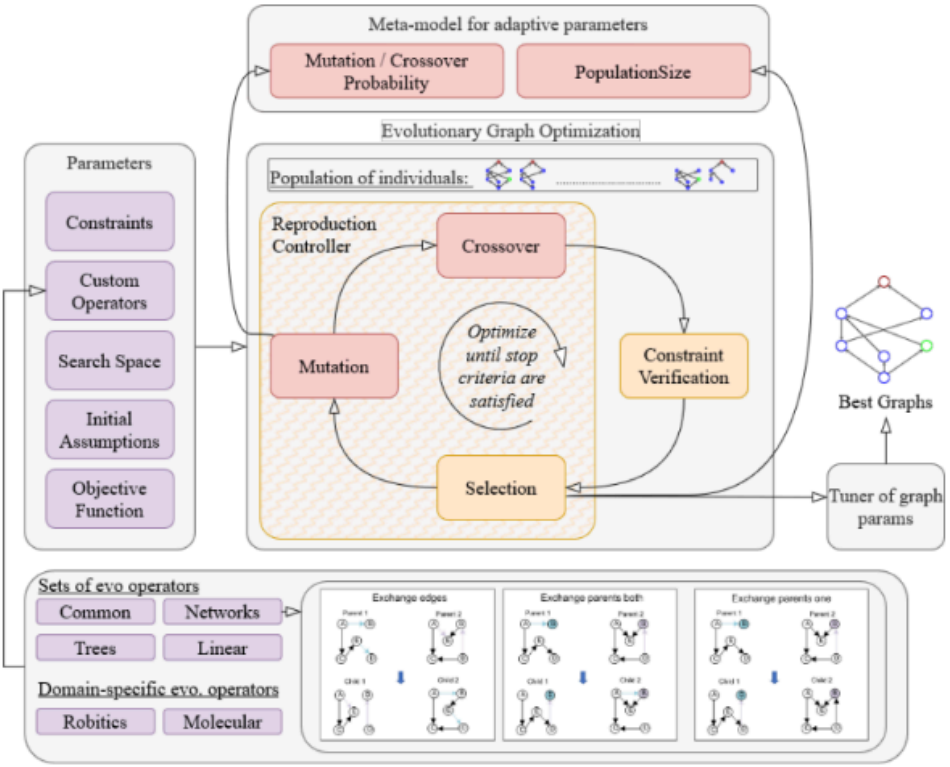

Рисунок 1. Схема структуры GOLEM: основные блоки и этапы

Мутации включают в себя базовые одноточечные модификации графа: добавление, удаление или замену случайно выбранного узла или ребра. Также реализованы сложные мутации роста и уменьшения для более быстрого исследования пространства поиска. Например, в качестве мутации может быть добавлен новый подграф. Одноточечные модификации графов делают набор мутаций более универсальным, а более сложные растущие и уменьшающие мутации позволяют быстрее исследовать пространство поиска.

4.2 Уровень адаптации домена

Для определения новой доменной задачи необходимо задать следующие отображения и параметры, в соответствии с постановкой задачи в разделе 2.3.

- G: пространство поиска графов с параметризованными узлами.
- F: минимизируемая целевая функция.
- A и R: отображения между структурами домена и представлением графа.
- C: набор ограничений (необязательно).

Главная особенность GOLEM - универсальность. Любая доменная структура, которая может быть преобразована в параметризованные графы, может быть оптимизирована с помощью GOLEM. Он обеспечивает гибкость в определении доменных структур.

Сопоставления Adapt и Restore позволяют естественным образом определить в доменных терминах цели, пользовательские операторы эволюции и пользовательские ограничения структуры. GOLEM применяет эти сопоставления для адаптации предоставленных функций стоимости и операторов домена к представлениям графов.
Например, если существует доменная цель F определенная на пространстве доменных структур $F : S \to R$, то можно получить цель F', определенную для графов: $F \circ A = F' : G \to R$. Аналогично, если существует доменная мутация $M : S \to S$, то можно получить графовую мутацию $R \circ M \circ A = M' : G \to G$. И F', и M' после описанных преобразований могут быть использованы в универсальном оптимизаторе графов. Аналогичная логика работает и для ограничений на доменные структуры. Самое главное, что эти преобразования применяются GOLEM автоматически. От пользователей требуется только предоставить отображения Adapt и Restore, реализовав интерфейс GraphAdapter с соответствующими методами adapt и restore.

5 ПРИМЕРОВ И ЭКСПЕРИМЕНТАЛЬНЫЕ ИССЛЕДОВАНИЯ

Большинство существующих бенчмарков, связанных с обучением графов, рассматривают исключительно задачи предсказания графов, узлов и связей (например, [29]) и не включают задачи поиска графов (в их нынешнем виде). Одними из немногих примеров подходящих решений являются однозадачный бенчмарк для проектирования лекарств [4] и более разнообразный бенчмарк для проектирования на основе моделей [42]. Мы объединили идеи существующих бенчмарков и поставили эксперименты на синтетических задачах поиска графов и реальных задачах в конкретных доменах.

5.1 Синтетические эксперименты

Для синтетических экспериментов использовались древовидные и DAG типы графов. Для генерации древовидных графов используется функция random_tree из NetworkX, однако для генерации DAG-графов сначала используется gnp_random_graph, а затем полученный граф подвергается постобработке путем удаления ребер, пока не станет направленным и ациклическим.

Метрики для сравнения графов использовались в соответствии с анализом в [48]. Для данного набора экспериментов использовались следующие метрики, характеризующие топологию графов:

- degree_dist - эвристическая метрика для графов, в которых центральные узлы более важны, чем периферийные. Чем выше степень узла, тем значительнее разница в количестве таких узлов между двумя графами. Таким образом, для каждого графа рассчитывается гистограмма степеней, а затем вычисляется расстояние между этими гистограммами. Эта метрика характеризует локальную топологию графа.

- sp_adj характеризует локальную и глобальную топологию графа и вычисляется как L2 расстояние между спектрами матриц смежности графов.

- graph_size - это разница между размерами графов. Она включена в эксперименты как простейшая базовая метрика.

Основное различие между сравниваемыми оптимизаторами заключается в том, что случайный оптимизатор итеративно применяет случайно выбранную мутацию для каждого графа в популяции, а эволюционный оптимизатор также использует эволюционный отбор на основе SPEA2 и одноточечный кроссинговер для каждой пары особей.

Каждый эксперимент был запущен 15 раз для 350 поколений. Для каждого типа графов были выбраны 50 и 100 целевых размеров графов. Результаты представлены в таблице 1. Эволюционный оптимизатор значительно превосходит случайный в подавляющем большинстве случаев. Кроме того, для выяснения статистической значимости полученных результатов был использован непараметрический тест Манна-Уитни, который подтвердил, что практически для всех наборов результаты статистически значимы.

metric	degree		graph_size		sp_adj	
Restoration error						
target	RS	Evo	RS	Evo	RS	Evo
d_50	0.72	**0.70**	24.72	**23.01**	4.94	**4.60**
d_100	**0.62**	0.65	94.55	**75.64**	18.91	**15.13**
t_50	0.37	**0.35**	10.49	**3.47**	2.10	**0.69**
t_100	**0.59**	0.80	51.95	**25.95**	10.39	**5.19**

Таблица 1: Сравнение качества восстановления и времени сходимости для базового метода случайного поиска (RS) и эволюционного оптимизатора в GOLEM. Лучшие результаты выделены жирным шрифтом.

Примеры графиков сходимости для метрики sp_adj и древовидных графов размера 100 представлены на рисунке 2а. Видно, что помимо того, что эволюционный оптимизатор превосходит случайный по метрикам, он еще и сходится на ранних этапах оптимизации, что делает его более оптимальным даже на запусках с меньшим числом поколений.

Основное преимущество фреймворка GOLEM заключается в том, что он применим к самым разным областям. Так, например, он используется в качестве ядра AutoML-фреймворка FEDOT. Однако в машинном обучении в большинстве случаев оптимальное решение лежит среди небольших конвейеров, состоящих из ML-моделей и операций предварительной обработки данных. Весь потенциал GOLEM раскрывается в конкретных задачах реального мира, представленных ниже.

5.2 Байесовские сети

В ходе анализа производительности GOLEM мы обратили внимание на задачу структурного обучения байесовских сетей (БС). Область эволюционных алгоритмов нашла заметное применение в контексте обучения структуры БС [25]. Тем не менее,

следует отметить, что преобладающие подходы были направлены на оптимизацию не самой графовой структуры, а ее кодированных представлений, таких как матрицы смежности [14], упорядоченные списки узлов [26] и т.д. Отличительной особенностью представленного решения на основе GOLEM является новый подход, позволяющий проводить эволюционную оптимизацию непосредственно структурной конфигурации БН.

5.2 Байесовские сети

При анализе производительности GOLEM мы обратили внимание на задачу обучения структуры Байесовских сетей (БС). Поле эволюционных алгоритмов видит заметное применение в контексте обучения структуры БС [25]. Однако стоит отметить, что преобладающие подходы сосредоточены не на оптимизации самой графической структуры, а на оптимизации представленных в ней кодированных представлений, таких как матрицы смежности [14], упорядоченные списки узлов [26] и т.д. Отличительной особенностью представленного решения на основе GOLEM является новая подходящая методика, которая позволяет осуществлять эволюционное оптимизирование прямо на конфигурации структуры БС.

Интеграция обучения структуры БС включала в себя объединение двух различных фреймворков: BAMT, специализирующегося на работе с Байесовскими сетями [7], и фреймворка GOLEM. В контексте фреймворка BAMT мы определили необходимые операции мутации и скрещивания, функцию оценки и правила выбора валидных структур. Для оценки использовалась функция K2, которая имеет довольно широкое применение в задачах обучения структур.

Для проведения сравнительного анализа результатов обучения структур были проведены серии экспериментов на заранее подготовленных наборах данных [38]. В качестве основного базового алгоритма был выбран алгоритм Greedy Hill Climbing. Результаты обучения оцениваются с двойной перспективой, используя два различных метрики: F1-оценка и значение функции оценки. В данном случае F1-оценка

показывает качество восстановления структуры (высокое значение означает большую схожесть структуры с исходной). В таблице 2 представлены результаты сравнения (средние для десяти запусков). Сравнительный анализ показал, что интегрированный в фреймворк GOLEM эволюционный оптимизатор эффективно решает задачу обучения структур для Байесовских сетей, демонстрируя превосходные результаты по сравнению с базовым алгоритмом. Однако для достижения результатов, превосходящих базовый уровень, требуется дальнейшее совершенствование на больших числах узлов.

Dataset	Evo	HC	Evo	HC
	F1		*Evaluation function*	
Asia (8)	**0.71**	0.29	**-2296**	-2300
Cancer (5)	**0.57**	0.57	**-2104**	-2105
Earthquake (5)	**1.00**	1.00	**-533**	-533
Sachs (11)	**0.62**	0.53	-7400	**-7391**
Sangiovese (15)	**0.2**	0.17	**-22835**	-22878
Mildew (35)	**0.28**	0.28	-52749	**-52076**
Barley (48)	**0.24**	0.23	-63835	**-61448**

Таблица 2: Сравнение результатов структурного обучения BN для эволюционного алгоритма (Evo) и базового жадного алгоритма (HC), для каждого набора данных в скобках указано количество узлов. Наилучшие результаты выделены жирным шрифтом.

Рисунок 2b иллюстрирует графики сходимости решений из десяти запусков, используя набор данных Sachs в качестве примера. Отметим, хотя решения ultimately сходим к одинаковым значениям функции оценки, стоит подчеркнуть, что GOLEM смог обнаружить превосходное решение при оценке через метрику F1. Это явление возникает из-за концепции эквивалентности в области Байесовских сетей. Суть заключается в том, что различные структуры сетей могут соответствовать одинаковым значениям функции оценки. Дано, что эволюционные алгоритмы интуитивно исследуют более широкий спектр решений, они имеют большее вероятность сходимости к желаемой ссылочной структуре. В противоположность, базовое решение обычно демонстрирует более детерминированный паттерн сходимости к постоянной структуре.

5.3 Разработка лекарств

В качестве еще одного случая для демонстрации универсальности и эффективности GOLEM мы выбрали задачу поиска лекарств. Целью является обнаружение молекул с определенным химическим свойством. Исследование огромного пространства молекул является сложной задачей, в основном из-за его огромного размера. Для решения проблемы целенаправленного генерирования молекул мы применили фреймворк GOLEM с использованием химической библиотеки RDKit.

Для получения новых молекулярных структур мы использовали предложенные в [27] пользовательские мутации. Реализованные операторы мутации можно разделить на две группы: первичные и вторичные. Первичные мутации представляют собой простые операции, такие как добавление, удаление или замена атома и удаление или замена связи. Более сложные, многократные действия соответствуют вторичным мутациям: удаление или перемещение функциональной группы, вставка углерода между двумя соединенными атомами (вставка углерода), удаление атома, если у него только два соседа (удаление атома).

Мы сравнили наш алгоритм с SMILES LSTM [39], Graph GA, CReM [36] и EvoMol [27] на бенчмарке GuacaMol [3]. Окончательные оценки ограничены интервалом [0, 1] (1 - лучшая оценка) и рассчитываются как взвешенные средние оценок молекулярных структур по таким характеристикам, как структурные особенности, физико-химические свойства, сходство или несходство с другими молекулами, наличие или отсутствие подструктур, функциональных групп или типов атомов. Результаты сравнения на 10 запусках представлены в таблице 3.

Для эксперимента были ограничены типы атомов до C, N, O, F, P, S, Cl и Br, а также использовались одинарные, двойные и тройные связи. Количество поколений установлено в 3000, размер популяции равен 50, а максимальное количество тяжелых атомов равно 50. Для генерации только возможных молекулярных структур был предложен набор возможных операций редактирования до применения мутаций. Для начальной популяции были выбраны лучшие 100 молекул по оценке из набора GuacaMol для каждого бенчмарка.

benchmark	SMILES LSTM	Graph GA	CReM	EvoMol	GOLEM	EvoMol best	GOLEM best
Celecoxib rediscovery	1.000	1.000	1.000	0.978	0.889±0.011	1.000	1.000
Troglitazone rediscovery	1.000	1.000	1.000	1.000	0.926±0.128	1.000	1.000
Thiothixene rediscovery	1.000	1.000	1.000	0.876	0.792±0.026	1.000	0.828
Aripiprazole similarity	1.000	1.000	1.000	1.000	1.000	1.000	1.000
Albuterol similarity	1.000	1.000	1.000	1.000	1.000	1.000	1.000
Mestranol similarity	1.000	1.000	1.000	1.000	0.999	1.000	1.000
C11H24	0.993	0.971	0.966	1.000	0.999	1.000	1.000
C9H10N2O2PF2Cl	0.879	0.982	0.940	0.998	1.000	1.000	1.000
Median molecules 1	0.438	0.406	0.371	0.455	0.382	0.455	0.388
Median molecules 2	0.422	0.432	0.434	0.417	0.405	0.417	0.405
Osimertinib MPO	0.907	0.953	0.995	0.955	0.946	0.969	0.955
Fexofenadine MPO	0.959	0.998	1.000	1.000	0.996	1.000	0.999
Ranolazine MPO	0.855	0.920	0.969	0.966	0.882±0.017	0.957	0.911
Perindopril MPO	0.808	0.792	0.815	0.845	0.751±0.032	0.827	0.782
Amlodipine MPO	0.894	0.894	0.902	0.867	0.918	0.869	0.920
Sitagliptin MPO	0.545	0.891	0.763	0.915	0.739±0.017	0.926	0.765
Zaleplon MPO	0.669	0.754	0.770	0.791	0.746	0.793	0.748
Valsartan SMARTS	0.978	0.990	0.994	0.998	0.988	0.998	0.991
Deco Hop	0.996	1.000	1.000	1.000	0.991	1.000	1.000
Scaffold Hop	0.998	1.000	1.000	1.000	0.984±0.016	1.000	1.000
total	17.341	17.983	17.919	18.061	17.330	18.211	17.693
total MPO	5.637	6.202	6.214	6.339	5.980	6.341	6.080

Таблица 3: Результаты теста с GuacaMol . Для оценки результатов используется пользовательский тестовый показатель, описанный в тексте.

5.4 Конструкция роботизированных манипуляторов

В данном случае GOLEM используется для исследования пространства поиска и нахождения почти оптимального конечного графа, описывающего механизм связей в роботизированной руке в библиотеке Rostok. Используя графовую представление, строится модель симуляции захвата, за которой следует моделирование процесса захвата объекта под влиянием внешних воздействий. Для симуляции жесткой динамики тел PyChrono [41] физический движок используется. Воздействие заключается в плавно увеличивающейся силе, направленной под углом 45 градусов к нормали ладони. На основе данных моделирования рассчитывается вознаграждение за оценку эффективности процесса хватания.

Результаты сравнения с другими алгоритмами представлены в таблице 4. GOLEM показывает высокую эффективность в поиске оптимальных решений, демонстрируя потенциал для применения в задачах проектирования роботизированных систем.

Оценка качества выполняется с использованием классических метрик захвата, включая долю фаланг, контактирующих с объектами, расстояние между центром объекта и геометрическим центром точек контакта, и способность выдерживать внешние силы, которые комбинируются в виде взвешенного суммирования. Значение вознаграждения примерно 9 указывает на успешный захват и удержание объекта.

Генетический алгоритм сам не учитывает ограничения на основе правил грамматики. Поэтому он может производить графы, представляющие невозможные механизмы. Для решения этой проблемы и улучшения эффективности алгоритма мы ввели две модификации: (1) вознаграждение за невозможный механизм установлено в ноль; (2) начальная популяция генерируется с использованием жизнеспособных дизайнов, полученных с помощью графической грамматики (GG). Начальная популяция формируется на основе предположения о разнообразии и включает кандидатов с 1, 2, 3 и 4 пальцами в равном количестве; (3) правило фильтрации рёбер для скрещивания, ограничивающее рёбра, не направленные к узлам "тела". Лучшие дизайны и график вознаграждения представлены на рисунке 3. Значения функции вознаграждения оцениваются с использованием симулятора.

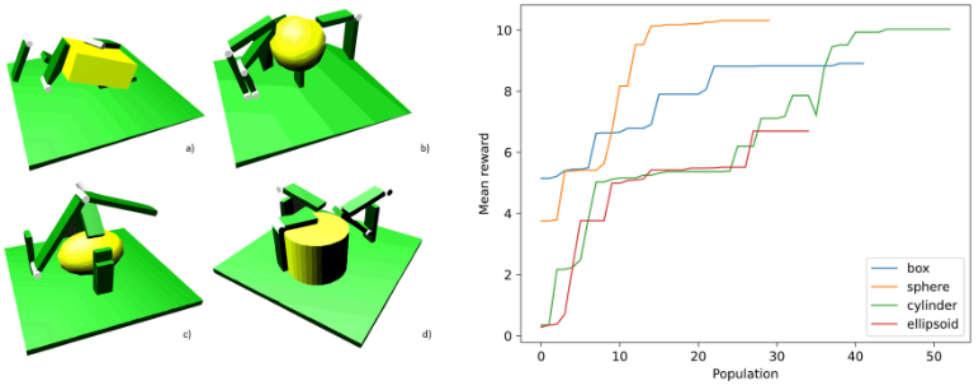

Рисунок 3: Слева. Захваты, созданные с помощью GOLEM framework: различные устройства для захвата таких объектов, как (а) коробка, (б) сфера, (в) эллипсоид, (г) цилиндр. Верно. Средняя физическая подготовка 3-х лучших особей в популяции показывает сходимость результатов оптимизации для каждой задачи.

Кроме того, мы применили Монте-Карло поиск в дереве (MCTS) и правила графической грамматики для решения той же задачи. Сравнение оценки вознаграждения представлено в таблице 4. Участие GOLEM улучшило результаты для 3 из 4 задач против MCTS.

Task	Box	Sphere	Cylinder	Ellipsoid
MCTS + GG	9.082	9.145	7.0	**10.019**
GOLEM	**11.259**	**10.331**	**10.116**	8.919

Таблица 4: Сравнение полученных вознаграждений эволюционным алгоритмом GOLEM и правилами грамматики графов, основанными на MCTS.

Таким образом, реализованный подход можно считать гибридом графовой грамматики и эволюционной оптимизации. Кроме того, мы расширили абстрактную реализацию генетического алгоритма, полученного от GOLEM, несколькими эволюционными операторами для конкретных задач (мутации и кроссинговер), которые специально разработаны для минимизации количества нежизнеспособных механизмов.

6 ВЫВОДЫ

В данной работе мы предложили гибкий и адаптивный подход под названием GOLEM для автоматизации задач проектирования в различных научных областях. Он обеспечивает модульный интерфейс, который учитывает целевые функции, специфичные для предметной области, пространство поиска и ограничения при использовании единого эволюционного ядра. Экспериментальная оценка GOLEM приведена для синтетических тестов и набора реальных примеров (разработка

вероятностных моделей, разработка лекарств, разработка роботизированных захватов). GOLEM доступен как инструмент с открытым исходным кодом, который может быть расширен для различных приложений.

Благодарности

Эта работа финансово поддерживается Фондом Национальной Технологической Инициативы в рамках проекта поддержки в рамках реализации дорожной карты развития высокотехнологичного направления Искусственный Интеллект до 2030 года (соглашение № 70-2021-00187).

Ссылки на литературу

[1] Лаит Мохаммад Абualiaга, Мохамед Э. Абд Элазиз, Ахмад М. Хасавне, Мохаммад Альшинван, Рехаб Али Ибрагим, Мохаммед Абдулазиз Айд Альканьс, Сейедали Мирджалили, Путра Сумари, Амир Хоссейн Гандоми. 2022. Алгоритмы метаэвристической оптимизации для решения задач реального машиностроения: обзор, приложения, сравнительный анализ и результаты. Neural Computing and Applications 34 (2022), 4081–4110. https://api.semanticscholar.org/CorpusID:247412220

[2] Джулиан Бланк и Калиймой Деб. 2020. Pymoo: Многокритериальная оптимизация на Python. IEEE Access 8 (2020), 89497–89509. https://api.semanticscholar.org/CorpusID:211076047

[3] Нейтан Браун, Марко Фискато, Марвин Г. С. Сеглер, Ален С. Вошер. 2019. GuacaMol: Бенчмаркинг моделей для де-ново молекулярного проектирования. Журнал химической информации и моделирования 59, 3 (2019), 1096–1108.

[4] Нейтан Браун, Марко Фискато, Марвин Г. С. Сеглер, Ален С. Вошер. 2018. GuacaMol: Бенчмаркинг моделей для де-ново молекулярного проектирования. Журнал химической информации и моделирования 59 3 (2018), 1096–1108. https://api.semanticscholar.org/CorpusID:53787096

[5] Ирене Кордоба, Эдуардо Г. Мерчан, Даниэль Х. Лобато, Конча Бьельца, Педро Ларранага. 2018. Байесовская оптимизация алгоритма PC для обучения байесовских сетей Гаусса. V Достижения в области искусственного интеллекта: 18-я конференция Испанской ассоциации искусственного интеллекта, CAEPIA 2018, Грана, Испания, 23-26 октября 2018 г., Материалы 18. Спрингер, 44-54.

[6] Дерек Г. Корнел и Ричард М. Крюгер. 2008. Обобщенная концепция поиска в графах. Журнал SIAM по дискретной математике 22, 4 (2008), 1259–1276.

[7] Ирина Деева, Анна Бубнова, Анна В. Калюжная. 2023. Расширенный подход к обучению параметров распределений в байесовских сетях с использованием моделей

смешанных распределений и дискриминативных моделей. Mathematics 11, 2 (2023), 343.

[8] Ци ци Дуань, Го Чен Чжоу, Чан Шао, Чжовэй Ван, Мин Ян Фэн, И-цзюнь Ян, Ци Чжао и Юхуэй Ши. 2022 год. PyPop7: Библиотека на чистом языке Python для оптимизации черного ящика на основе популяции. arXiv abs / 2212.05652 (2022). https://api.semanticscholar.org/CorpusID:2545642488

[9] Томас Эльскен, Ян Хендрек Метцен, Франк Хюттер. 2019. Поиск нейронных архитектур: обзор. Журнал исследований в области машинного обучения 20, 1 (2019), 1997–2017.

[10] Маттиас Фей и Ян Ленссен. 2019. Быстрое обучение представлений графов с использованием PyTorch Geometric. В Семинар ICLR по изучению представлений на графах и многообразиях.

[11] Джонатан Годвин, Томас Кек, Питер Баттаглия, Виктор Бапст, Томас Кипф, Юйя Ли, Кимберли Стэченфельд, Петар Величкович, Альваро Санчес-Гонсалес. 2020. Jraph: библиотека для графовых нейронных сетей в JAX. http://github.com/deepmind/jraph

[12] Нихил Гойял, Харш Вардхан Джайн, Саян Рану. 2020. Graphgen: масштабируемый подход к домен-агностическому генерированию меток графов. V Материалы веб-конференции 2020. 1253-1263.

[13] Джорджо Грицети, Райнер Кюммерле, Хаук Страсдат, Курт Конолиге. 2011. g2o: общая рамка для (гипер)графовой оптимизации. В Материалы Международной конференции IEEE по робототехнике и автоматизации (ICRA). 9–13.

[14] Д. Ханцелка. 2008. Использование гибридных генетических алгоритмов в обучении структуры байесовских сетей из данных. Журнал прикладной математики 1, 2 (2008), 387–396.

[15] Син Хе, Каййонг Чжоу, Сяовен Чу. 2021. AutoML: обзор состояния искусственного интеллекта. Knowledge-Based Systems 212 (2021), 106622.

[16] Юваль Хеффец, Роман Вайнштейн, Гилад Кэтц, Лир Роках. 2020. Deepline: инструмент AutoML для генерации пайплайнов с использованием глубокого машинного обучения и фильтрации иерархических действий. В Материалы 26-й международной конференции ACM SIGKDD по поиску знаний и интеллектуальному анализу данных. 2103–2113.

[17] Э. Хенальт, Мария Харрис Рассмуссен, Ян Х. Йенсен. 2020. Исследование химического пространства: как генетические алгоритмы находят иглу в стоге.Простая физическая химия (2020). https://api.semanticscholar.org/CorpusID:225459180

[18] Эмили С. Хенальт, Мария Х. Рассмуссен, Ян Х. Йенсен. 2020. Исследование химического пространства: как генетические алгоритмы находят иглу в стоге. PeerJ Physical Chemistry 2 (2020), e11.

[19] Макдональд Дж. 2014. Руководство по статистике в биологии.

[20] Дежун Цзян, Чжэнсинь Ву, Чанью Хси, Гуаньюн Чен, Б. Ляо, Чжоу Ван, Чао Шен, Дуншен Цао, Цзянь Ву, Тингжун Хоу. 2020. Может ли обучение графовых нейронных сетей лучше представлять молекулярные структуры для поиска лекарств? Сравнительное исследование моделей на основе описателей и графовых моделей. Журнал химиоинформатики 13 (2020). https://api.semanticscholar.org/CorpusID:231940292

[21] Джон Джампер, Ричард Эванс, Александр Прицель, Тим Грин, Майкл Фигурнов, Олаф Роннебергер, Кимберли Стэченфельд, Петар Величкович, Альваро Санчес-Гонсалес. 2021. Высокоточное предсказание структуры белка с помощью AlphaFold. Nature 596, 7873 (2021), 583–589.

[22] Элиас Халил, Ханьюн Дай, Юйю Чжан, Бистра Дилкина, Ле Сонг. 2

[23] Амит Дилип Кини, Сварадж Самбхаджи Ядав, Адитья Шанкар Тхакур, Акшар Баджранг Авари, Цименг Лю, Тревис Деселл. 2023. Совместно развивающиеся рекуррентные нейронные сети и их гиперпараметры с оптимизацией симплексных гиперпараметров. В Материалы Международной конференции по генетическим и эволюционным вычислениям. 1639–1647.

[24] Невилл Китсон, Антоний Константиноу, Жигао Гуо, Ян Лью, Киатикун Чобтхам. 2023. Обзор структуры обучения Байесовских сетей. Обзор искусственного интеллекта (2023), 1-94.

[25] Педро Ларранага, Хоссейн Каршенас, Конча Бьельца, Роберто Сантьяго. 2013. Обзор эволюционных алгоритмов в задачах обучения и инференции Байесовских сетей. Информационные науки 233 (2013), 109–125.

[26] Чжун Хун Ли, Ву Йонг Чунг, Энтай Ким. 2008. Обучение структуры Байесовских сетей с использованием двойного генетического алгоритма. IEICE transactions по информации и системам 91, 1 (2008), 32-43.

[27] Юль Леги, Томас Коши, Марта Глвашкых, Беатрис Дювал, Бенуа Да Мота. 2020. EvoMol: гибкий и интерпретируемый эволюционный алгоритм для

неупорядоченного генерирования молекул. Журнал химиоинформатики № 12 (2020).https://api.semanticscholar.org/CorpusID:221716089

28 Юоужи Лю, Кэкван Ян, Шуиван Джи. 2021. GraphDF: Дискретный потоковый модель для генерации молекулярных графов. В International Conference on Machine Learning. https://api.semanticscholar.org/CorpusID:231749761

[29] Кристофер Моррис, Нилс М. Криеге, Франка Баузе, Кристиан Керстинг, Петра Мутцел, Марьон Нойман. 2020. TUDataset: коллекция бенчмарков для обучения с графами. ArXiv abs/2007.08663 (2020). https://api.semanticscholar.org/CorpusID:220633407

[30] Варнавас Д. Моухлис, Антреас Афантис, Анджела Серра, Микеле Фрателло, Анастасиос Г. Пападиамантис, Василис Аидинис, Изолт Линч, Дарио Греко, Джорджия Мелаграки. 2021. Прогресс в де-новом дизайне лекарств: от традиционных к методам машинного обучения. Международный журнал молекулярных наук 22, 4 (2021), 1676.

[31] Кирилл В. Насонов, Дмитрий В. Иволга, Иван И. Борисов, Сергей А. Колюбин. 2023. Компьютерное проектирование закрытых цепных механизмов: робот, управляемый морфологическим вычислением. V В 2023 году состоится Международная конференция IEEE по робототехнике и автоматизации (ICRA). IEEE, 7419-7425.

[32] Ченхао Ниу, Ян Сонг, Джяминг Сонг, Шенджя Жао, Адитья Гровер, Стефано Эрмон. 2020. Генерация инвариантных графов с использованием моделирование оценок. V Международная конференция по искусственному интеллекту и статистике. PMLR, 4474-4484.

[33] Бабатунде Моктард Олуоладе, Джелиан Гао, Джямин Чен, Тенгфей Лю, Рейед Аль-Сабри. 2022. Поиск архитектуры графовых нейронных сетей: обзор. Tsinghua Science and Technology (2022). https://api.semanticscholar.org/CorpusID:245115768

[34] Дамиан Оверко, Фернандо Гама, Алейандро Рибейро. 2019. Оптимальное управление потоком мощности с использованием графовых нейронных сетей. ICASSP 2020 - Международная конференция IEEE 2020 по акустике, речи и обработке сигналов(ICASSP) (2019), 5930–5934. https://api.semanticscholar.org/CorpusID:204823873

[35] Юнь Пенг, Байрон Чой, Джелианг Сюй. 2021. Обучение графов для комбинаторной оптимизации: обзор состояния искусственного интеллекта. Data Science and Engineering 6, 2 (2021), 119–141.

[36] Павел Полышчука. 2020. CReM: рамка для химически разумных мутаций для генерации структур. Journal of Cheminformatics 12, 1 (2020), 1–18.

[37] Тим Салиманс, Джонатан Хо, Си Чен, Илья Сутскевер. 2017. Стратегии эволюции как масштабируемый альтернативный подход к обучению с подкреплением. ArXiv abs/1703.03864 (2017). https://api.semanticscholar.org/CorpusID:11410889

[38] Марко Скутари. 2017. Алгоритмы обучения структуры Байесовских сетей с использованием ограничений: параллельные и оптимизированные реализации в пакете bnlearn R. Journal of Statistical Software 77, 2 (2017), 1–20. https://doi.org/10.18637/jss.v077.i02

[39] Марвин ГС Сеглер, Тири Когей, Кристиан Тирхан, Марк П. Уоллер. 2018. Генерация фокусированных библиотек молекул для поиска лекарств с использованием рекуррентных нейронных сетей. ACS Central Science 4, 1 (2018), 120–131.

[40] Никита О. Стародубцев, Николай О. Никитин, Елизавета А. Андронова, Константин Г. Гаваза, Денис О. Сидоренко, Анна В. Калюжная. 2023. Генеративное проектирование физических объектов с использованием модульного фреймворка. Engineering Applications of Artificial Intelligence 119 (2023), 105715.

[41] Алессандро Тасора, Раду Сербан, Хаммад Мазхар, Арман Пазуки, Даниэль Меланц, Джонатан Флайшманн, Майкл Тейлор, Хироюки Сугияма, Дэн Негрута. 2016. Chrono: открытая многофизическая динамическая система. В High Performance Computing in Science and Engineering: Second International Conference, HPCSE 2015, Soláň, Czech Republic, May 25-28, 2015, Revised Selected Papers 2. Springer, 19–49.

[42] Брэндон Трабукко, Синьянг Гэнг, Авирам Кумар, Сергей Левин. 2022. Design-bench: Бенчмарки для данных-ориентированного офлайн моделирования. В International Conference on Machine Learning. PMLR, 21658–21676.

[43] Михаил Цагрис. 2022. Алгоритм обучения Байесовских сетей FEDHC. Mathematics 10, 15 (2022), 2604.

[44] Патрик Уолтерс и Регина Барзилай. 2020. Применение глубокого обучения в генерации молекул и предсказании молекулярных свойств. Accounts of Chemical Research 54, 2 (2020), 263–270.

[45] Миньцзе Ванг, Да Чжэн, Цихао Е, Кван Ган, Мудей Ли, Сян Сонг, Цзинцзин Чжоу, Чао Ма, Лингфан Ю, Ю Гай, Тяньюн Сяо, Тонг Хэ, Джордж Карипис, Джинянг Ли, и Чжэн Чжан. 2019. Deep Graph Library: Граф-центрированная, высокопроизводительная библиотека для графовых нейронных сетей. arXiv preprint arXiv:1909.01315.8), 31–36.

[48] Питер Уиллс и Франсуа Г. Мейер. 2019. Метрики сравнения графов: Практическое руководство. PLoS ONE 15 (2019). https://api.semanticscholar.org/CorpusID:118711105

[49] Сонг Сяндуо, Ван Син, Сонг Юйюань, Цзоу Сянлин, и Ван Ин. 2022. Иерархические рекуррентные нейронные сети для генерации графов. Information Sciences 589 (2022), 250–264.

[50] Сюнь Чжэн, Брайон Арагам, Прадип К. Равикумар и Эрик П. Син. 2018. Развертки без разрывов: непрерывная оптимизация для изучения структуры. Достижения в области нейронных систем обработки информации, 31 (2018).

[51] Давэй Чжоу, Личэн Чжан, Цзеджун Сюй и Цзинжуй Хэ. 2019. Разное:
Многомасштабная порождающая модель для графиков. Рубежи в области больших данных 2 (2019), 3.