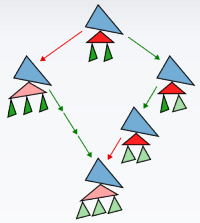
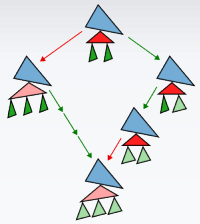


Лабораторная работа 1: Неуловимый-Джо

Использование больших языковых моделей для подсказчика по ТФЯ:

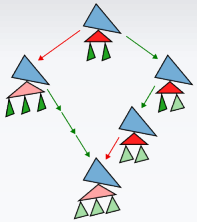
- Разработка чат-бота по теоретической части ТФЯ с подключением собственной базы знаний.
- Разработка фреймворка для решения задач по ТФЯ с испытанием на интерактивном прувере завершаемости систем переписывания термов.



Чат-бот по теории

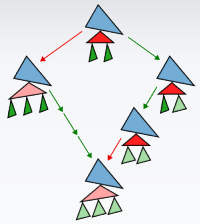
- База знаний в формате "запрос-ответ"векторизуется и разбивается на chunks (библиотечными средствами, библиотека на выбор).
- По запросу, вводимому пользователем, строится контекст из наиболее близких элементов базы знаний, после чего запрос вместе с контекстом передаётся LLM для построения ответа.

Фрейм чат-бота: 1 человек
Заполнение базы знаний: ∞ человек. С каждого минимум по 12 элементов базы. Желательно разделить темы по исполнителям.



Фреймворк для решения задач

- Фреймворк принимает грамматику (фиксированную, см. следующий слайд). Пользователь вводит в чат запрос на проверку интерпретации завершимости TRS в свободной форме (примерно так: «*дана система переписывания термов: такая-то, и я интерпретирую её конструкторы такими-то функциями. Доказывает ли моя интерпретация завершаемость trs?*»). Пользователь не обязан точно следовать синтаксису, приведённому в грамматике, и не обязан точно следовать шаблону выше.
- Посредством подходящего промта требуется, чтобы LLM переформулировала этот запрос в подходящую для формального разбора форму (метод Formalize). Допускается, чтобы на этом этапе форма также не полностью совпадала с определяемой грамматикой, но чтобы из неё уже легко было извлечь таковую формальными средствами (метод Convert).
- Результат метода Convert передаётся в метод Parse (см. далее).



Фреймворк для решения задач-2

Грамматика для TRS следующая:

variables = ([буква],)* [буква]

([терм] = [терм][eol])+

Здесь [eol] — перевод строки, перевод каретки, или совместно и то и другое. Нетерминал [терм] — составной терм из конструкторов и переменных. Все конструкторы имеют имена из одной буквы. Все буквы без аргументов, не являющиеся переменными, считаются нульместными конструкторами (константами).

Интерпретацию и грамматику можно хранить в разных потоках, а можно в одном, используя разделитель вида ‘————‘ (или любой другой).

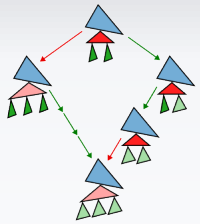
Грамматика для интерпретации следующая:

([конструктор]([переменная],)* [переменная]) = [моном](+[моном])*[eol])+

[константа] = [коэффициент]

[моном] ::= (([коэффициент]*)?[переменная]({[степень]})?)+|[коэффициент]

Степень и коэффициент — натуральное число (не ноль). Переменные в мономе могут повторяться. Задача метода Parse — перевести TRS и интерпретацию во внутреннюю форму, пригодную для конвертации в модель SMT, либо вывести сообщение об ошибке.



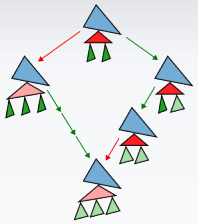
Обработка сообщений об ошибках

Минимальный список сообщений об ошибках:

- Неправильная скобочная структура в правиле переписывания
- Некорректный полином (синтаксис полинома не соответствует описанию выше)
- Не хватает интерпретаций для всех конструкторов trs (исключая одноместные)
- Повторные или избыточные интерпретации для конструкторов trs

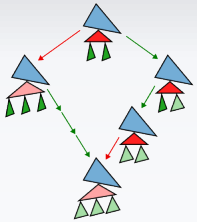
Сообщение об ошибке выдаётся пользователю. Если пользователь решает, что проблема не на его стороне, он должен сообщить об этом чату, и на основе сообщения об ошибке и исходного промта делается запрос к LLM по переформулировке задачи (запускается новый цикл Formalize -> Parse).

Если ошибок не обнаружено, результат формализации, соответствующий грамматике, также выводится пользователю на оценку. Если пользователь подтверждает корректность грамматики, вызывается метод Interpret, иначе пользователь должен сам указать на ошибку, и начинается новый цикл Formalize -> Parse.



Интерпретация в SMT-решателе и демо

- Переменные в правилах переписывания разделяются: в каждом правиле вводятся собственные переменные. Например, если дана trs $f(x) = g(x)$, $g(x) = f(f(x))$, то она приводится к виду $f(x_1) = g(x_1)$, $g(x_2) = f(f(x_2))$. Все переменные объявляются в заголовке SMT-файла.
- Мономы упрощаются, приводятся подобные, полиномы подставляются в левые и правые части всех правил TRS.
- Строятся условия существования контрпримера: хотя бы в одном правиле левая часть меньше либо равна правой.
- Соответствующая задача передаётся SMT-решателю. Если он возвращает ответ unsat, тогда пользователю сообщается об успехе верификации. Иначе пользователю передаётся контрпример, построенный решателем.
- Если верификация успешна, то также делается демо: строятся случайные термы в сигнатуре TRS без переменных и делается несколько случайных шагов переписывания, после чего подсчитывается вес исходного терма в интерпретации и полученного после переписывания.



Распределение

- Formalize+Convert — 2-3 человека
- Parse — 2-3 человека (на 3 человека — более точные и расширенные сообщения об ошибках)
- Interpret — 2 человека.