



Санкт-Петербургский государственный университет
Кафедра системного программирования

Параллельная реализация miniKanren

Андрей Антонович Диденко, 241 группа

Научный руководитель: Д.С.Косарев, ассистент кафедры системного программирования

Санкт-Петербург
2022

- Работа направлена на повышение эффективности вычислений на языке miniKanren путем декомпозиции задачи на потоках
- Данное решение предназначено для разработчиков, использующих miniKanren
- Сообщество minikanren всерьез еще этим не занималось, поэтому у работы нет аналогов

Используемые инструменты, подходы

- За основу взят язык программирования OCaml, а также встраиваемый в него язык miniKanren
- Для разработки моего проекта была задействована библиотека Domainslib для OCaml версии 5
- В 5 версии добавлены инструменты для параллелизации с более удобным функционалом

Целью работы является распараллеливание miniKanren

Задачи:

- Выбрать версию, на которой будет реализована параллельность
- Научиться параллелить две независимые задачи на miniKanren
- Подзадачи: запустить параллельно `appendo`, `reverso`

- Выбрана самая свежая версия OCaml (5.0.0 beta2) ¹ В этой версии добавили multicore в runtime и с помощью этого легко параллелить программы на OCaml
- Почему была выбрана 5 версия? В 4 версии нет настоящего параллелизма (приходится запускать системные потоки и параллельность происходит на уровне системы, а не на уровне OCaml)

¹<https://ocaml.org/news/ocaml-5.0.beta2>(Дата доступа: 08.12.22)

- Для распараллеливания используется библиотека DomainsLib ² В которой присутствуют 2 модуля: Task - для вызова многопоточности и Chan - для передачи информации между доменами
- Для освоения способов применения я изучил предложенные примеры параллельного сложения матриц и также рассмотрел реализацию чисел Фибоначчи, которая тоже была распараллелина

²<https://github.com/ocaml-multicore/parallel-programming-in-multicore-ocaml/blob/master/README.md>(Дата доступа: 08.12.22)

- Вся проделанная работа выполнена на минимальной реализации miniKanren - Unicanren³, которая состоит из 4 базовых конструкций: Fresh, Unify, Conde и Conj
- В этой работе есть возможность параллелить только дизъюнкцию (conde), так как это выполнение двух независимых задач, в то время как все остальное - независимые задачи и нет осознания того как их параллелить

³<https://github.com/Kakadu/unicanren>(Дата доступа: 08.12.22)

- Так как для меня не только miniKanren, но и язык OCaml оказались незведанными, мне пришлось помучиться для их освоения
- Я начал с простых задач:
 - ▶ Параллельный запуск appendo
 - ▶ Параллельный запуск reverso (при котором возникла проблема нехватки памяти)
 - ▶ Параллельный запуск некоторых функций, которые возвращают несколько ответов
 - ▶ Последней задачей оказалось слияние Stream'ов: Хотим доставать ответы по мере поступления изнутри функции и вытягивать их на верхний уровень(в ответ)

Сделал много подготовительной работы. Поэтому могу приступить к нормальной реализации интерпретатора miniKanren на параллельном OCaml

Добавление параллелизации - это важная новость в мире OCaml, острее науки об этом пишут статьи, поэтому тема является неизученной и актуальной для дальнейших улучшений miniKanren

- Выбрал 5 версию OCaml для реализации
- Научился параллельно запускать 2 независимые задачи
- Подготовился к реализации настоящего интерпретатора⁴

⁴<https://github.com/K0lba/unicanren> (Дата доступа: 07.12.22)