Санкт-Петербургский государственный университет

Кафедра системного программирования

Параллельная реализация miniKanren

Андрей Антонович Диденко, Б-07.21 ММ

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

Санкт-Петербург
2022

Введение

- 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

Постановка задачи

Целью работы является распараллеливание miniKanren
Задачи:

- Выбрать версию OCaml, на которой будет реализована параллельность
- Научиться параллелить две независимые задачи на miniKanren
- Подзадачи:
  - запустить параллельно appendo
  - запустить параллельно reverso
  - Объединить поток Stream в ответ

Используемые инструменты, подходы

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

OCaml

- 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)

DomainsLib

- 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 распараллелена

[2] https://github.com/ocaml-multicore/parallel-programming-in-multicore-ocaml (Дата доступа: 08.12.22)

Unicanren

- Вся проделанная работа выполнена на минимальной реализации miniKanren – Unicanren[3], 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 - зависимые задачи и нет осознания того, как их параллелить

[3] https://github.com/Kakadu/unicanren(Дата доступа: 08.12.22)

Процесс работы

- 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 поступаления изнутри функции и вытягивать их на верхний уровень(в ответ)

Дальнейшие планы

Сделать параллельно исполняющуюся реализацию miniKanren

Результаты

- Выбрал 5 версию OCaml для реализации
- Научился параллельно запускать 2 независимые задачи[4]
- Подготовился к реализации настоящего интерпретатора[4]

[4] `https://github.com/K0lba/unicanren` (Дата доступа: 07.12.22)