Санкт-Петербургский государственный университет

Кафедра системного программирования

Группа 21.Б07-мм

Автоматизированное построение расписания занятий в вузах на miniKanren

Габдрахманов Азат Райнурович

Отчёт по учебной практике
в форме «Решение»

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

Санкт-Петербург
2023

Оглавление

Введение

В наше время до сих пор остается актуальной проблема составления расписания. В середине прошлого века начался бурный рост теории расписаний [3]. Задачи теории расписаний связаны с упорядочиванием некоторых работ по времени и/или исполнителям с учетом ограничений [5]. Одной из задач теории расписаний является задача построения расписания для учебных заведений.

В следующих предложениях слово “расписание” будет использоваться в контексте расписания для высших учебных заведений. В наше время расписание часто составляется вручную. При составлении расписания следует учитывать ряд формальных ограничений, таких как, например, невозможность проводить одному преподавателю два занятия одновременно, невозможность проводить лекционное занятие в обычном классе и другие. Также при составлении расписания желательно учесть и множество других аспектов. Слишком большая кучность занятий увеличивает утомляемость учащихся. Следует учитывать и пожелания преподавателей.

Можно учитывать множество этих факторов вручную, но это может быть сопряжено с некоторыми трудностями. Например, если диспетчер расписания на некотором этапе не сможет корректно поставить занятие, то локальное перестроение расписания может привести к кардинальному изменению всего расписания, что усложняет работу.

Задаче составления расписания было уделено внимание со стороны научного сообщества и были разработаны различные методы поиска решения. Учитывая ограничения, включая NP-полноту задачи, хорошо себя показывают эвристические методы, такие как имитация отжига и эволюционные алгоритмы [5]. Также широкое распространение для решения задач теории расписаний получил метод программирования в ограничениях [3].

MiniKanren — семейство встраеваемых языков реляционного программирования, который ранее не использовался при решении данной задачи, если судить по открытым источникам. MiniKanren является

языком программирования в ограничениях, поэтому хотелось бы раскрыть потенциал полезного в академическом плане языка при решении практической задачи. В рамках данной работы предлагается разработать решение задачи автоматического построения расписания в вузах с помощью языка реляционного программирования miniKanren.

Дата сборки: 31 мая 2023 г.

1. Постановка задачи

Целью работы является создание web-приложения с возможностью автоматического построения расписания занятий в вузах с помощью miniKanren со следующими начальными данными:

- учебный план всех групп;
- количество аудиторий и их специализация;
- данные педагогической нагрузки;
- набор ограничений.

Для её выполнения были поставлены следующие задачи.

Осенний семестр

1. Разработать процедуру поиска расписания на miniKanren без параметризации, соответствующего следующим ограничениям:
   - учет всех предметов из учебного плана;
   - проверка аудиторий на специализацию.
2. Визуализировать вручную ответ в виде одной общей таблицы с расписанием.

Весенний семестр

1. Параметризация ввода учебного плана.
2. Параметризация ограничений.
3. Создать web-приложение, в котором можно вызвать составление расписания и получить его.

2. Обзор

В данном разделе будут рассмотрены готовые продукты для составления расписания. Обзор существующих методов решения задач теории расписания, а не продуктов, можно увидеть в работе [3].

2.1. Обзор решений

Для автоматического составления расписания реализованы различные решения. В открытом доступе затруднительно найти информацию об алгоритмах составления расписания данных решений. В данном разделе рассмотрим продукты с точки зрения получения расписания. Данные о решениях собраны из различных открытых источников, основной из них [4].

- БИТ.ВУЗ.Расписание[1].
- 1С:Автоматизированное составление расписания.Университет[2].
- Экспресс-расписание Колледж[3].
- Система «АВТОРасписание»[4].
- Расписание занятий: «Ректор-Колледж»[5].

Рассмотрим лидирующие в области решения.

2.1.1. БИТ.ВУЗ.Расписание

У решения есть полная и “lite” версии, рассмотрим полную:
Преимущества данного решения:

- возможность изменять расписание с помощью таблиц;

[1] https://spb.1cbit.ru/1csoft/bit-vuz-raspisanie/, Дата доступа: 30.05.2023
[2] https://solutions.1c.ru/catalog/asp_univer/materials, Дата доступа: 30.05.2023
[3] https://pbprog.ru/docs/raspis/, Дата доступа: 30.05.2023
[4] https://www.mmis.ru/programs/avtor, Дата доступа: 30.05.2023
[5] https://rector.spb.ru/, Дата доступа: 30.05.2023

- возможность учесть/игнорировать множество ограничений;
- мониторинг ошибок при составлении;
- функция составления расписания в смешанном режиме.

Недостатки:

- возможность составить расписание автоматически только для одной группы за раз;
- нет возможности использовать бесплатно.

2.1.2. 1С:Автоматизированное составление расписания

Преимущества данного решения:

- возможность изменять расписание с помощью таблиц;
- возможность выбора периодичности расписания (неделя,две недели и т.д.);
- возможность учесть/игнорировать множество ограничений;
- возможность установить ограничения по переходам между корпусами и аудиториями;
- мониторинг ошибок при составлении;
- функция составления расписания в смешанном режиме;
- удобный WEB-сервис.

Недостатки:

- нет возможности использовать бесплатно;
- до покупки затруднительно найти исчерпывающую информацию об алгоритме и о возможностях программы для автоматического построения расписания.

Вывод

Хотелось бы создать продукт с ясным алгоритмом составления расписания и возможностью составлять расписания на несколько потоков.

2.2. Используемые инструменты

2.2.1. miniKanren

miniKanren[6] — семейство встраеваемых языков реляционного программирования. В работе будет использоваться OCanren[7] — реализация miniKanren на OCaml.

Основные определения и понятия. Более подробную информацию по miniKanren можно увидеть в статье [2].

Программы на miniKanren пишутся с помощью реляций (от англ. “relation” — отношение). Реляция — вычислимое отношение. Ядром miniKanren в языке OCaml являются несколько конструкций: $cond^e$, $\equiv$ (введено вместо $===$), $fresh$, &&& (конъюнкция)

Конструкция $\equiv$ называется унификацией и используется для того, чтобы показать, что два аргумента являются одинаковыми.

Листинг 1: Пример применения унификации

```
let _ = run_exn 1 q qh (REPR (fun q → (≡ q 5))
// (5)
let _ = run_exn 1 q qh (REPR (fun q → (≡ q 5) &&&
(≡ q 6))
// ()
```

Оператор run служит интерфейсом между OCaml и miniKanren, а переменная, объявленная вторым аргументом функции run_exn, указывает программе, что нужно вывести ее значение. Первая программа присваивает q значение 5, но вторая не выдает ответа, так как нет такого значения q, которое ровно одновременно и 5, и 6. Число после run означает количество первых ответов, которые требуется получить от программы. Но в данном случае, если даже вместо 1 будет -1 (что говорит програм-

[6] http://minikanren.org/, Дата доступа: 30.05.2023

[7] https://github.com/PLTools/OCanren, Дата доступа: 30.05.2023

ме вывести все имеющиеся ответы), то вывод у программы будет тот же. Конструкция &&& принимает две реляции и заканчивается успешно только когда две реляции могут одновременно завершиться успешно. Иначе ответа не будет.

Конструкция $cond^e$ зачастую используется для получения нескольких ответов. Рассмотрим применение $cond^e$ на примере.

Листинг 2: Пример применения $cond^e$

```
(let _ = run_exn (-1) q qh (REPR (fun q →
        (cond^e
           [(≡ q '(matan))] // 1 ветка
           [(≡ q '(alg))] // 2 ветка
           [(≡ q '(geom))] // 3 ветка
           )) ⟹ ((matan) (alg) (geom))
```

Получены ответы (matan) (alg) (geom), это означает, что любой из трех списков удовлетворяет реляции $cond^e$. Реляция $cond^e$ “удовлетворена”, точнее считается завершенной успешно, если успешна завершается хотя бы одна ветвь $cond^e$.

Конструкция $fresh$ используется для создания новых переменных.

Листинг 3: Использование $fresh$ и унификации

```
let _ = run_exn 1 q qh (REPR (fun q → (fresh (x y) (≡ x y)
(≡ x 3) (≡ y q))))
// (3)
```

Таким образом, в данной программе x, y и q равны 3 и переменные перестают быть новыми, теперь попытка унификации любой из переменных с любым значением, кроме 3, приведет к пустому выводу программы.

2.2.2. OCaml

OCaml[8] — объектно-ориентированный язык функционального программирования общего назначения[9]. В OCaml нашими соотечественниками реализован miniKanren, называемый OCanren.

[8] https://ocaml.org/, Дата доступа: 30.05.2023

[9] https://ru.wikipedia.org/wiki/OCaml, Дата доступа: 30.05.2023

2.2.3. Js_of_Ocaml

Js_of_Ocaml[10] — компилятор программ байт-кода OCaml JavaScript. Компилятор сначала преобразует байт-код в промежуточное представление SSA, для которого выполняется оптимизация, прежде чем генерировать JavaScript[1].

[10] http://ocsigen.org/js_of_ocaml/latest/manual/overview, Дата доступа: 30.05.2023

3. Реализация

3.1. Реализация поиска расписания, соответствующего требованиям

На вход дается учебный план в виде нескольких строк, вид которых указан в README файле[11].

3.1.1. Представление данных

В данной работе расписание каждой группы или преподавателя представляет собой список из пяти списков, где каждый отдельный список представляет собой один день недели, в котором хранится четыре строки - каждая из которых отвечает за одну из четырех пар.

Пары в расписания ставятся последовательно, поэтому нужно хранить промежуточные расписания. Они хранятся в переменной, представляющей собой список пар. Ключ - название группы или имя преподавателя, значение - соответствующее расписание. Эта переменная является реляционной, но minikanren не умеет реляционно обрабатывать сложные структуры данных. Из-за этого тратится линейное время на поиск значения по ключу. В секции 3.2 будет показан вариант частичного решения проблемы.

3.1.2. Вставка предметов в расписание преподавателя и группы

Особенностью miniKanren является то, что при составлении запроса важен порядок конъюнктов. Таким образом, нужно понять, при каком порядке запрос будет завершен быстрее всего.

Важные частные случаи.

1. Обычно группы более заняты, чем преподаватели, поэтому конъюнкт, который отвечает за вставку предмета в группу, ставится раньше, чем конъюнкт, отвечающий за преподавателя.

[11] https://github.com/Azatic/schedule/blob/githubpages/README.md, Дата доступа: 30.05.2023

2. Как говорилось выше, отдельно на вход подаются планы лекций и практик, поэтому нужно понять, в каком порядке лучше разместить конъюнкты, отвечающие за лекции и за практики.

   Дать универсальный ответ сложно, ведь он зависит от входных данных. Поэтому сравниваются лекции и практики, и в зависимости от этого ставятся конъюнкты. Хотя это тоже не всегда приводит к ускорению, в некоторых ситуациях сложно предсказать поведение miniKanren'a.

Ниже будет приведен пример реляции, которая используется для вставки предметов в расписание преподавателя и группы. Далее английское слово "slot" будет использоваться как перевод слова "пара".

Листинг 4: Реляция, использующая унификацию для вставки 1 пары. Реляция принимает расписание на определенный день у группы, преподавателя и аудитории и вставляет в этот день на первую пару предмет.

```
  let insert_subj_on_first_slot subj group_schedule teacher_schedule  =
    fresh ()
      (group_schedule ≡ Std.list Fun.id [ subj; __; __; __ ])
      (teacher_schedule ≡ Std.list Fun.id [ subj; __; __; __ ])
;;
```

Если все три переменные, в которые мы вставляем предмет, новые («свежие»), то реляция завершится успешно. Если хотя бы одна из переменных занята, то есть в чьем-то расписании 1 пара уже занята, то реляция не завершается и программа будет пытаться вставить предмет уже в другую пару или день. Символы __ означают, что в это некоторые переменные, которые реляция не меняет.

3.1.3. Ограничения

В работе реализовано два ограничения:

- Возможность вставки окон в определенный день.
- Возможность разным группам присутствовать на одной практике.

Возможность вставки окон в определенный день.

Листинг 5: Реляция принимает на вход день, номер и расписание и вставляет окно на эту пару.

```
let delete_day n q group =
  conde
    [ n ≡ !!''monday'' &&& delete_monday q group
    ; n ≡ !!''tuesday'' &&& delete_tuesday q group
    ; n ≡ !!''wednesday'' &&& delete_wednesday q group
    ; n ≡ !!''thursday'' &&& delete_thursday q group
    ; n ≡ !!''friday'' &&& delete_friday q group
    ]
;;
```

Возможность разным группам присутствовать на одной практике.

Бывает ситуация, когда у двух групп есть практики, например, по программированию, но студенты одной группы, хотят так же присутствовать на паре другой группы. Поэтому, когда у первой группы будет стоять программирование, у другой будет стоять окно

3.2. Ускорение алгоритма

На протяжении всей работы программы все расписания хранятся в переменной «storage», которая представляет собой список пар: ключ - название группы или имя преподавателя, значение - расписание соответсвующей группы или преподавателя. Из-за того, что при вставке каждого предмета нужно искать в этой переменной и расписание преподавателя, и расписание группы, время программы растет. Поэтому перед началом работы программы все имена преподавателей и названия групп заменяются целыми числами, так как сравнение чисел быстрее сравнения строк.

3.3. Дополнительные возможности

3.3.1. Кучность

Возможность "сдвигать" пары относительно дней недели или номера пар. То есть miniKanren может сперва ставить с первой по четвертую пару в понедельник, затем во вторник и так далее. Можно так же ставить в первую очередь, например, 3 и 4 пары понедельника, то же самое у вторника и так далее, а первые две пары заниматься будут в последнюю очередь.

Листинг 6: Реляция, представляющая собой вставку пар, которая принимает на вход расписание преподавателя, группы и аудитории и предмет. В первой ветке реляция пытается вставить предмет на первую пару понедельника у группы и преподавателя, во второй - на вторую пару понедельника и т.д.

```
let insert_all_sched subj group_sched teacher_sched
class_sched =
fresh
  (a1 a2 a3 a4 a5 b1 b2 b3 b4 b5 c1 c2 c3 c4 c5)
  (group_sched ≡ Std.list Fun.id [ a1; a2; a3; a4; a5 ])
  (teacher_sched ≡ Std.list Fun.id [ b1; b2; b3; b4; b5 ])
  (class_sched ≡ Std.list Fun.id [ c1; c2; c3; c4; c5 ])
  (conde
     [ insert_lesson_to_first_pair subj a1 b1 c1
     ; insert_lesson_to_second_pair subj a1 b1 c1
     ; insert_lesson_to_third_pair subj a1 b1 c1
     ; insert_lesson_to_second_pair subj a1 b1 c1
     ; insert_lesson_to_first_pair subj a2 b2 c2
     ; insert_lesson_to_second_pair subj a2 b2 c2
     ; insert_lesson_to_third_pair subj a2 b2 c2
     ; insert_lesson_to_second_pair subj a2 b2 c2
     ])
     ;;
```

Представлены только первые 8 строк реляции.

3.4. Визуализация

расписание группы/преподавателя b-07

	Понедельник	Вторник	Среда	Четверг	Пятница
9:30 - 11:05	окно				
11:15 - 12:50	окно	математический_анализ_1	математический_анализ_2		
13:50 - 15:15	окно	алгебра_1			
15:25 - 17:00	окно	Математический_анализ_лекция_1			

Если на месте пары стоит пустая ячейка, то это значит, что в это время группа свободна. В случае, если в ячейке стоит слово «окно», это значит, что в это время у группы точно не будет занятия.

4. Замеры производительности

Замеры проводятся на ноутбуке MacBook Air 13 M1, 8 Gb RAM. Расписание строилось на основе учебного плана потока «Программная инженерия» на два курса без учета английского языка и онлайн курсов. Всего 4 группы и 14 преподавателей.

Было произведено 10 замеров, их результаты:

(101.061461, 100.168143, 102.090688, 100.573399, 101.654358, 101.304482, 100.879553, 101.195086,101.153573)

Погрешность = 0,4

Среднее = 101,1

Стандартное отклонение среднего = 0.2

Тестовое покрытие

Тестовое покрытие составляет 84%.

5. Дальнейшие планы

Не исключено продолжение данной работы, поэтому была проведена оценка сложности реализаций следующих ограничений.

5.1. Отсутствие окон

В miniKanren каждой переменной сопоставляется некоторое особое число, называемое номером переменной. Каждой паре в расписании занятий соответствует некоторая переменная с уникальным номером. При выводе программы можно сразу видеть, что переменная осталась ”свежей”, то есть она не имеет значение, что в нашем случае значит пустую пару. Но на этапе работы программы не так просто понять, является ли переменная свободной, или нет. Для реализации функции проверки на окна понадобится использовать нереляционные конструкции.

5.2. Единственная пара за день

При решении вышепоставленной задачи решение этой задачи следует почти автоматически.

Научившись проверять переменные на ”пустоту”, нам останется написать реляцию, которая, принимая расписание на день недели определенной группы , будет заканчивать провально при единственной непустой переменной, и успешно в остальных случаях.

6. Результаты

На данный момент выполнены следующие задачи:

1. Параметризация процедуры поиска учебного плана.
2. Параметризация ограничений.
3. Визуализация.

Код проекта доступен в GitHub-репозитории:
https://github.com/Azatic/schedule
Web-приложение доступно по ссылке:
https://azatic.github.io/schedule/

Список литературы

[1] Balat Vincent. Js_of_ocaml. — URL: https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=1b87d68c7c4f7b897eeb09804657225f6f8f762f (дата обращения: 2022-12-14).

[2] William E. Byrd Eric Holk Daniel P. Friedman. miniKanren, Live and Untagged. — URL: http://webyrd.net/quines/quines.pdf (дата обращения: 2022-12-14).

[3] Лазарев Александр Алексеевич Гафаров Евгений Рашидович. Теория расписаний, задачи и алгоритмы. — URL: https://www.ipu.ru/sites/default/files/publications/12896/406-12896.pdf (дата обращения: 2022-12-14).

[4] Н.В. Самсонова. Составление расписания в высшем учебном заведении: математические методы и программные продукты. — URL: https://guu.ru/wp-content/uploads/Самсонова-Н.В.-Симонов-А.Б..pdf (дата обращения: 2022-12-14).

[5] Н.Н. Клеванский. Алгоритмы формирования расписания занятий высших учебных заведений. — URL: https://fundamental-research.ru/ru/article/view?id=41857 (дата обращения: 2022-12-14).