Задания к спецкурсу «Алгоритмы дискретной оптимизации»
Адрианов Н.М., Иванов А.Б. // май 2021

Ниже приведен список заданий, которые можно решать получения зачета по ЕНС. Какие-то задачи можно делать теоретически (доказать утверждение, придумать кодирование...), какие-то подразумевают программное решение с использованием SAT- или LP-солвера (предлагаем использовать python + пакеты PuLP и PySAT), либо реализации другого алгоритма на языке по выбору.

Воспринимайте этот список как ориентир. Если в задаче несколько пунктов, это не означает, что надо делать их все. Они там потому что есть некоторая идея, которую можно развивать. Приветствуется, если вы какие-то свои идеи развития (сравнить скорость решения выбранной вами задачи с помощью SAT и LP, найти интересные наборы экземпляров задач (например, есть библиотеки судоку, задач раскраски, задач рюкзака), применить идеи из нашего спецкурса в других задачах, головоломках или играх.

PuLP: `https://coin-or.github.io/pulp/`

PySAT: `https://pysathq.github.io/`

При использовании SAT в каких-то приложениях может понадобиться реализация «псевдобулевых» ограничений типа $x_1 + \ldots + x_n \leq r$ – для булевских переменных x_i мы требуем, чтобы не менее r из них были истинны и записываем это как арифметическое выражение, интерпретируя их как $0/1$. Есть разные техники кодирования таких условий в виде КНФ, они доступны в PySAT: `https://pysathq.github.io/docs/html/api/pb.html`

1. Хроматическое число. Дан граф $G = (V, E)$ и целое число k. Раскрасить вершины в k цветов так, чтобы никакие две вершины, соединенные ребром, не оказались окрашены в одинаковый цвет.

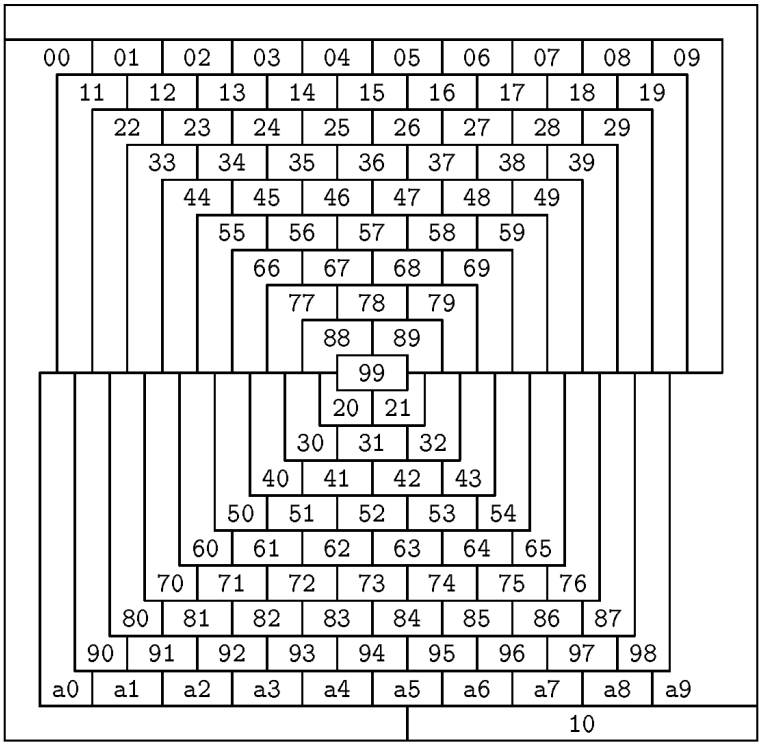
Для формулы 3-SAT на $n \geq 4$ переменных $x_1, \ldots x_n$, состоящей из клауз $C_1, \ldots C_m$, определим граф $G = (V, E)$ так:

$$V = \{x_1, \ldots x_n\} \cup \{\bar{x}_1, \ldots \bar{x}_n\} \cup \{v_1, \ldots v_n\} \cup \{C_1, \ldots C_m\}.$$

$$\begin{array}{rcccccc} E & = & \{(x_i, \bar{x}_i) \mid i = 1, \ldots n\} & \cup & \{(v_i, v_j) \mid i \neq j\} & \cup \\ & & \{(v_i, x_j) \mid i \neq j\} & \cup & \{(v_i, \bar{x}_j) \mid i \neq j\} & \cup \\ & & \{(x_i, C_k) \mid x_i \notin C_k\} & \cup & \{(\bar{x}_i, C_k) \mid \bar{x}_i \notin C_k\}. & \end{array}$$

Нарисуйте описанный граф и убедитесь, что это сведение задачи 3-SAT к задаче о хроматическом числе. Какое количество цветов нужно потребовать в этой задаче? Зачем в этой конструкции ограничения 3-SAT (а не просто SAT) и $n \geq 4$?

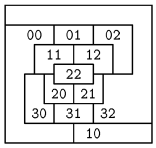
2. Раскрасьте в 4 цвета карту, опубликованную Мартином Гарднером в качестве первоапрельского розыгрыша:



(а) Руками.

(б) Используйте SAT- или LP-солвер для раскраски этой карты. Файл с описанием графа для раскраски в формате DIMACS приложен.

Мартин Гарднер использовал карту, придуманную Вильямом Макгрегором. Эту карту можно включить в семейство карт разного размера. Например, карта Макгрегора порядка 3 будет иметь следующий вид:



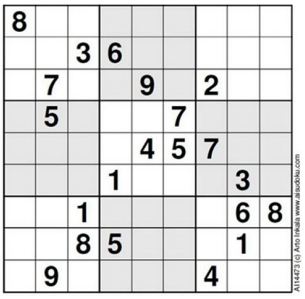
(в) Опишите конструкцию карты Макгрегора порядка n. Сколько вершин и ребер в соответствующем (двойственном) графе? Какие степени вершин?

(г) Есть ли 4-клики в графах Макгрегора? Сколько их?

(д) Найдите раскраски карт Макгрегора небольшого размера так, чтобы количество стран, окрашенных в некоторый фиксированный цвет, было как можно меньше. Существуют раскраски, в которых некоторый цвет используется для не более чем $5n/6$ стран. Можно ли конструкцию этих раскрасок распространить на произвольные n?

(е) Используйте SAT-солвер или LP-солвер, чтобы найти минимальное r такое, что существует раскраска карты Макгрегора порядка n такая, что количество стран, окрашенных в один из цветов равно r (если будете использовать SAT – потребуется записать «псевдобулево» ограничение).

3. Решите «самую сложную головоломку судоку» (по мнению *The Telegraph*):



Напишите программу, решающую произвольные судоку с помощью SAT или LP. Легче руками или написать программу?

4. Придумайте способ решения японского кроссворда с помощью сведения к ILP (это совсем не так тривиально, как судоку). Если у вас есть другие любимые логические головоломки – попробуйте свести их к SAT/ILP.

5. Известен следующий приближенный алгоритм решения задачи минимального вершинного покрытия:

(а) Сводим задачу к ЦЛП: вводим переменные $x_u = 0/1$ для каждой вершины $u \in V$ и записываем ограничения $x_u + x_v \geq 1$ для каждого ребра $(u, v) \in E$. $x = (x_u)_{u \in V}$ – некоторое оптимальное решение.

(б) Зафиксируем значения x_u, которые равны 0, 1/2 или 1. Покажите, что остальные п.

(в) Покажите, как из точного полуцелого решения получить целое решение не более чем в 2 раза хуже оптимального.

(г) Предположим, что вершины графа раскрашены в 4 цвета. Выполните более аккуратный переход к целому решению с использованием раскраски вершин и покажите, что полученное решение не более чем в 3/2 раза хуже оптимального.

5. Решите задачу Киркмана о 15 школьницах: "Пятнадцать молодых девушек в школе прогуливаются по три в ряд семь дней (каждый день), требуется распределить их на каждую прогулку так, чтобы никакие две девушки не шли в том же ряду".

6. Задача Киркмана – вариация на тему блок-дизайнов. (v, b, r, k, λ)-блок-дизайном называется множество $X = \{1, \dots v\}$, и семейство его подмножеств $\mathcal{B} = \{B_i\}_{i=1}^{b}$, называемых блоками, такие, что каждый блок состоит ровно из r точек, каждая точка содержится ровно в k блоках, каждая пара точек содержится ровно в λ блоках.

Решение задачи Киркмана дает $(15, 35, 3, 7, 1)$-блок-дизайн.

Один из вопросов теории блок-дизайнов – вопрос существования блок-дизайна с заданными параметрами. Легко показывается, что параметры долны удовлетворять условиям $vk = br$ и $\lambda(v-1) = k(r-1)$, однако эти соотношения недостаточны.

Если $v = b$ и $r = k$, то блок-дизайн называется симметрическим (v, r, λ)-блок-дизайном. Симметрический блок-дизайн с параметром $\lambda = 1$ называется конечной проективной плоскостью. Проективные плоскости над конечным полем из q элементов дают примеры $(q^2 + q + 1, q + 1, 1)$-блок-дизайнов, однако существуют и другие проективные плоскости. Доказано, параметры любой конечной проективной плоскости имеют вид $(n^2+n+1, n+1, 1)$, n называется порядком проективной плоскости. Предполагается, что не существует конечных простых плоскостей порядка n, если n – не степень простого (гипотеза недоказана). Доказано, что не существует конечных плоскостей порядка $n = 6, 10$.

Сформулируйте задачу поиска проективной плоскости в виде SAT или LP. Докажите с использованием солвера, что не существует конечной проективной плоскости порядка $n = 6$. Удастся ли найти таким методом «грубой силы» плоскости порядков $n = 7, 8, 9$? Получится ли доказать, что не существует конечной проективной плоскости порядка $n = 10$?

7. Решите задачу Эйлера о 36 офицерах: "Разместить 36 офицеров шести различных полков и шести различных воинских званий в каре так, чтобы в каждой колонне и в каждом ряду был ровно один офицер каждого полка и каждого воинского звания.".

Задача Эйлера о 36 офицерах – занимательная формулировка вопроса о существовании пары ортогональных латинских квадратов порядка 6. Латинским квадратом порядка n называется матрица размера $n \times n$, состоящая из чисел от 1 до n, такая, что в каждой строке и каждом столбце все числа различны (например, в судоку мы заполняем латинский квадрат). Пара латинских квадратов $A = (a_{ij})$ и $B = (b_{ij})$ называется ортогональной, если все пары (a_{ij}, b_{ij}) различны.

а) Сам Эйлер решить эту задачу не смог (отрицательный ответ получен только в 1901 году). Сформулируйте задач в виде SAT или LP и решите ее.

В 1938 году показали, что из существования конечной проективной плоскости порядка n следует существование $n - 1$ попарно ортогональных латинских квадратов порядка n. Таким образом из отрицательного решения задачи Эйлера следует не существование конечной проективной плоскости порядка 6.

б) Эйлер ошибочно предполагал, что ортогональных латинских квадратов не существует для всех $n = 4k + 2$. В 1959 году обнаружили пару ортогональных квадратов порядка 10. Очень хорошее описание истории `http://math4school.ru/greko_latinskie_kvadrati.html`.

Получится ли SAT/LP-солвером найти пару квадратов порядка 10?

8. Задача о рюкзаке. На странице `http://hjemmesider.diku.dk/~pisinger/codes.html` выложены наборы тестовых задач (small coefficients, large coefficients, hard instances).

(а) У нас есть целый арсенал алгоритмов для решения задачи о рюкзаке: жадный, два метода динамического программирования, приближенный алгоритм на основе динамики, сведение к LP. Попробуйте реализовать и сравнить какие-то из этих алгоритмов.

(б) Продумайте/реализуйте branch-and-bound (branch-and-cut) алгоритм для задаче о рюкзаке. В основе таких алгоритмов – перебор вариантов по дереву (ветвление - берем предмет k или нет). Обязательным компонентом является возможность относительно легко получить оценку оптимального решения подзадачи, которая используется для отсечения ветвей дерева. В

случае ЦЛП – такая оценка получалась из решения релаксированной задачи ЛП. С рюкзаком даже проще – оценкой может служить оптимальное решение задачи о дробном рюкзаке, который очень быстро решается жадным алгоритмом.

(в) На странице `http://hjemmesider.diku.dk/~pisinger/codes.html` упомянуты и более продвинутые алгоритмы. Выберите какой-нибудь из алгоритмов и разберите/реализуйте его.

9. Генетические алгоритмы. Попробуйте применить их к какой-нибудь задаче (например, к задаче о рюкзаке). Реализация генетического алгоритма поднимает много вопросов – как выполнять скрещивание и мутацию, как отбирать "выживших тут очень большая свобода в реализации.