


Правительство Российской Федерации
Федеральное государственное автономное
образовательное учреждение высшего образования
«Национальный исследовательский университет «Высшая школа экономики»

Факультет компьютерных наук
Образовательная программа бакалавриата 01.03.02 «Прикладная математика и
информатика»

ОТЧЕТ
по учебной практике

на факультете компьютерных наук НИУ ВШЭ
(название организации, предприятия)

Выполнил(а) студент(ка)
группы БПМИ209-1
Дистлер М. А.
(инициалы, фамилия)

(подпись)

Руководитель практики

Департамент больших данных и информационного поиска / Международная лаборатория
теоретической информатики, ведущий научный сотрудник
(подразделение ФКН, должность)

Гурвич Владимир Александрович
(ФИО руководителя практики)
Дата 01/09/2021 10 (десять) Вури
(оценка) (подпись)

Москва – 2021

Оглавление

1.	Аннотация	3
2.	Цель и задачи практики	4
3.	Описание решения	6
5.	Анализ результатов	9
6.	Список использованных источников	10
7.	Выводы	11

1. Аннотация

Описываю, как я относилась к практике и как это отношение менялось для того, чтобы обосновать неактуальность плана-графика и индивидуального задания.

Я проходила практику на ФКН. Изначально было заявлено, что на практике будет около 3 человек и будут решаться только игры Секи и Д-Секи. Исходя из этого я и написала индивидуальное задание и план-график. Но мои ожидания не оправдались. На практике было 12 человек, и мы решали не только Секи и Д-Секи, но и игры Одномастка Вист, Одномастка Дурак, Одномастка Д-Дурак и НИМ на гиперграфах. Каждый студент решал только некоторую часть задач, все работали в одиночку или в небольших командах над своей узкой независимой областью, не пересекающейся с чужими задачами. Именно поэтому то, что я писала в индивидуальном задании и плане-графике, по независящим от меня причинам, является неактуальным.

2. Цель и задачи практики

Студентам был предоставлен выбор из нескольких игр, алгоритм решения которых нужно было написать и задачей создания сайта, который будет объединять все эти игры. Я решила заняться играми Одномастка Дурак и Одномастка Д-Дурак. Сначала стоит напомнить правила этих игр.

Правила Одномастки Дурак:

Пусть числа 1, 2, ..., x написаны на карточках и каким-то образом поделены между двумя игроками (сданы). У игроков может быть разное число карт на руках. Назовем это распределение карт позицией в игре. Удобно задавать ее с помощью 01-вектора длины x, нули и единицы которого соответствуют картам игроков 0 и 1.

0	1	1	0	0	0	1	1	1	1
1	2	3	4	5	6	7	8	9	10

Например, этот вектор означает, что игроку 0 принадлежат карты номер 1, 4, 5, 6, а игроку 1 карты 2, 3, 7, 8, 9, 10.

Указано, кто из игроков ходит первым. Без нарушения общности будем считать, что 0, так как в ином случае просто поменяем все 1 на 0 и 0 на 1 (то есть перенумеруем игроков) и будем считать, что начинает игрок 0. Он выбирает одну из своих карт и выкладывает ее на стол. Игрок 1 может:

1. Забрать карту себе, далее это называется принять. После этого игрок 0 ходит опять.
2. Побить, положив карту старше. Тогда сыгранные две карты выбывают из игры и ходит игрок 1.

Тот, у кого первым не осталось карт – выиграл со счетом в количество оставшихся карт противника. Игроки стремятся получить как можно больший счет, то есть сделать так, чтобы у них закончились карты, а у противника при этом осталось их как можно больше. Соответственно проигрывающий игрок будет стремиться уменьшить количество карт, оставшихся на руках на момент проигрыша.

Правила Одномастки Д-Дурак:

Правила этой игры отличаются от предыдущей только в одном нюансе: если у противников карты кончились во время одного хода, то засчитывается ничья. То есть в позиции 01, если первым ходит игрок 0, то в Д-Дурак игра закончится ничьей (при оптимальной игре), а в Дурак выиграет игрок 0 со счетом 0. По сути, ничья в Д-Дурак совершенно аналогична ситуации в Дурак, когда один из игроков выиграл со счетом 0.

Модификация Одномастки Дурак и Д-Дурак с весами:

Кроме игр с обычными правилами, которые я расписала выше, мне было предложено написать программу, которая будет решать ту же задачу для модификации Одномастки Дурак и Д-Дурак с весами.

Ее основное отличие от предыдущих правил состоит в том, что каждой карте в начале игры присваивается вес – любое целое число. Также меняются правила окончания игры. В этой модификации счет, с которым закончилась игра – это не количество оставшихся у одного из игроков карт, а сумма весов этих карт. Когда у одного из игроков закончились карты, то он выигрывает, только если сумма весов всех карт, оставшихся у другого игрока, больше или равна 0 (если это Д-Дурак, то счет 0 означает ничью). В ином случае, то есть если сумма весов карт, оставшихся у другого игрока, отрицательна, то выигрывает игрок, у которого остались карты. При этом игроки стремятся максимизировать сумму весов оставшихся у противника карт, на момент, когда наши карты кончились; и соответственно минимизировать сумму весов оставшихся у себя карт, на момент, когда карты противника кончились.

Задача практики:

Вернемся к самой цели практики. Передо мной была поставлена задача написать программу, которая по позиции в игре и тому, какой игрок ходит первым, а также весам карт, если это модификация игры с весами, должна определить:

1. Кто выиграет (при условии, что игроки ходят оптимально)
2. С каким счетом (при условии, что игроки ходят оптимально)
3. Все оптимальные ходы для текущей позиции для игрока, ходящего первым
4. “Хитрые” ходы, а именно:
 - a. Ловлю взятие. То есть минимальную карту, которую противник должен принять, а побить которую — это ошибка.
 - b. Ловлю пропускание. То есть максимальную карту, которую противник должен побить, а принять - ошибка.
5. Оптимальный ответный ход ходящего вторым игрока для любого варианта хода первого игрока (то есть принять карту или побить, и если побить, то какой картой)

3. Описание решения

Перед тем, как приступить к написанию алгоритма, я нашла информацию в интернете о комбинаторных играх, а также прослушала онлайн-лекции, которые нам зачитал Владимир Александрович Гурвич.

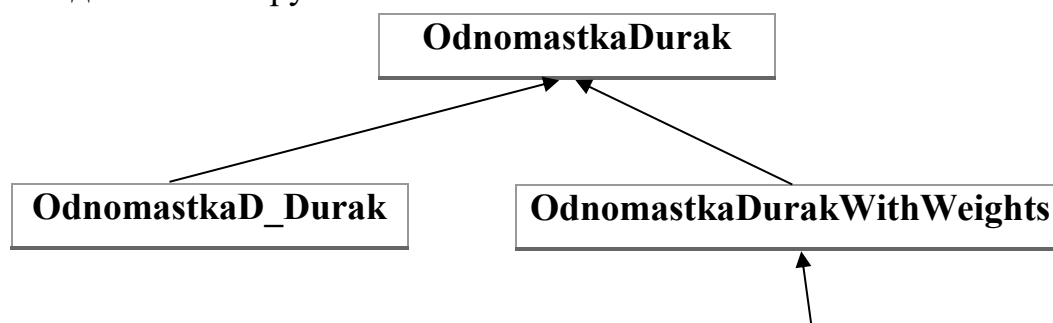
1. Общий алгоритм

Решения, которое я реализовала, основано на обратной индукции и рекурсии. То есть чтобы определить все данные, которые требуются, я сначала рассматриваю начальную позицию, потом перебираю все позиции, которые игрок 0 может получить после своего хода, для каждой из этих позиций, я перебираю позиции, которые получатся после ответного хода игрока 1 и так далее. Если после ответного хода должен ходить игрок 1, то я переименую игроков, то есть меняю все 1 на 0 и 0 на 1, и далее считаю, что ходит игрок 0. Так как игра не может быть бесконечной, то в каждой последовательности позиций мы придем к ситуации, когда у одного из игроков карты кончились. Для такой позиции очевидно, кто выиграет и с каким счетом. Зная это, я могу определить эту информацию и для позиции, из которой я попала в эту и так далее. Чтобы найти оптимальные и “хитрые” ходы, я просто сравниваю во время индукции, какие ходы привели к лучшим результатам, и запоминаю это. Каждой позиции, в которую я попадаю в процессе перебора, в поле класса игры (тип: словаре) сопоставлен объект класса **Position**, в котором и записаны все полученные результаты (а именно кто выиграл, с каким счетом, оптимальные и “хитрые” ходы, оптимальные ответные ходы).

Стоит упомянуть один важный нюанс: в игре без модификаций учитывалось только 2 варианта ответных хода – принять или побить минимально возможной картой. Мы не доказывали на практике, почему ходить в ответ оптимально только минимально возможной картой, но в программе реализован именно такой вариант. В реализации игры с модификацией такого нет, то есть в ответ перебираются не только варианты принять или побить, но и варианты какой картой побить.

2. Особенности реализации

Чтобы реализовать этот алгоритм для разных игр, я решила написать несколько классов. Я привожу схему наследования классов, задающих каждый свою игру:



OdnomastkaD_DurakWithWeights

Также я уже упоминала класс **Position**, который хранит данные, которые мы должны найти.

Отличий классов между собой довольно мало. Классы игр Дурак и Д-Дурак (неважно с модификацией или без) отличаются только тем, что происходит, если игра закончилась со счетом 0. В Дураке кто-то выигрывает, а в Д-Дураке ничья. Между Одномасткой с и без модификаций отличий больше. В игре с модификацией нужно учитывать веса карт, и потому позицией считаются не только распределение карт между игроками, но и веса этих карт. Также различие находится в том, что в игре без модификации счет – это просто количество карт противника при конце игры, а в игре с модификацией – это сумма весов этих карт, которую надо также проверить, является ли она отрицательной. Игра без модификации – это частный случай игры с модификацией, когда веса всех карт равны 1, но так как вариант с весами работает медленнее, то я вынесла его в отдельный класс. В общем-то, сам алгоритм во всех этих играх полностью одинаковый, меняются только нюансы реализации.

Также я хочу упомянуть некоторые решения, которые я использовала в реализации и которые кажутся мне интересными:

- В игре без модификации при построении, я считаю позиции 01..0, 010.. и 0.1.0 (где . означает уже сыгранные карты) совершенно одинаковыми и равными вектору 010. Это существенно убыстряет работу программы, так как уменьшается количество вариантов. К сожалению, в игре с модификациями эта идея не работает так хорошо, так как эти позиции не будут давать одинаковый результат, если карты в них имеют разный вес.
- Позицию, то есть 01-вектор, я хранила не в виде вектора, а в виде числа, где его двоичное представление соответствует данному вектору, к которому в конец приписали 1 (это позволяет отличить позиции 01 и 0100 между собой). Такой способ хранения помогает гораздо быстрее удалять и добавлять элементы, а также переименовывать игроков с помощью бинарных операций.
- В коде достаточно часто используются различные степени 2, поэтому я просчитала их заранее.

Кроме основного алгоритма я реализовала функцию, которая делает самый оптимальный ход (если их несколько, то один из “хитрых”) и переводит класс в новую позицию, а также функцию, которая по номеру карты ходит этой картой. То есть функции, которые ходят за компьютер или за игрока.

3. Код

Ссылка на репозиторий с кодом (все в файле main.py):

https://github.com/MarinaDistler/hse_practice_2021/tree/main

5. Анализ результатов

В результате практики я написала алгоритм, которые с помощью обратной индукции решает для небольшого количества карт игры Одномастка Дурак и Д-Дурак с весами или без. Игра без весов работает за приемлемое время (до 3 секунд) при количестве карт до 20, а игра с весами при количестве до 12. В результате работы алгоритма по данной начальной позиции можно узнать, кто выиграет и с каким счетом при оптимальной игре обоих игроков, оптимальные и “хитрые” ходы игрока 0 в текущей позиции, а также оптимальные ответные ходы игрока 1 на любой из возможных ходов игрока 0.

6. Список использованных источников

- Онлайн-лекции Владимира Александровича Гурвича.
- Википедия: [сайт]. – URL: https://ru.wikipedia.org/wiki/%D0%97%D0%B0%D0%B3%D0%BB%D0%B0%D0%B2%D0%BD%D0%B0%D1%8F_%D1%81%D1%82%D1%80%D0%B0%D0%BD%D0%B8%D1%86%D0%B0.
- PythonRu: [сайт]. – URL: <https://pythonru.com/>.
- DevPractice: [сайт]. – URL: <https://devpractice.ru/>.

7. Выводы

Я немного научилась работать с людьми и получила больше опыта в этом деле, улучшила некоторые свои навыки и знания в программировании, а также теории игр.

Что еще можно было бы сделать:

- Модифицировать, ускорять алгоритм Одномастки.
- Создать другой алгоритм, не решающий игру полностью, а угадывающий верный ход, пусть менее точно, зато быстрее и для более большого количества карт.
- Создать базу данных для решенных векторов.