Содержание

Аннотация

В проекте рассматриваются конечные графовые игры с n учасниками. Такие игры моделируются ориентированными графами, возможно с ориентированными циклами, однако мы считаем попадание в цикл, как конечный исход. Проект основан на статьях: [4] и [3]. В основном проект состоит в проверке гипотез выведенных из статьи [3] и описаных в [5]:

1 не существует конфигураций где нет равновесия Нэша и цикл является худшим из исходов для всех игроков (в неравенстве в примере на 4 странице [3])

2 не существует конфигураций где нет равновесия Нэша и цикл является 2-ым худшым из исходов хотя-бы для всех кроме 1-го игроков

3 не существует конфигураций бесповторной[1] игры без равновесия Нэша

Помимо существуют и другие гипотезы о играх без равновесия Нэша которые хочется проверить.

Ключевые слова

Теория игр, равновесие Нэша, графовые игры

[1]Каждый игрок контролирует только одну позицию. При этом положение цикла не имеет значения.

1 Введение

В 1950 году Нэш опубликовал свою фундаментальную работу где обобщил Minimax/saddle points позиции на игры с произвольным числом игроков. [7] В этой работе я рассмотрю детерминированные игры на графах с чистыми стратегиями и большим числом участников. Конкретнее найду примеры таких игр где нет равновесий Нэша (как на примере на 4 странице [3]) и проверю на этих примерах catch22 из [5]. Я буду искать игры которые опровергают catch22 или хотя-бы не содержат равновесия Нэша для проверки других гипотез. Для этого я буду пытаться построить игры на произвольных неизоморфных графах и проверять отсутствие в них равновесий Нэша. Конкретнее о том как я это делал и какие эвристики мне помогли далее в отчете.

2 Базовые определения

В 1990 году Уошберн [8] определил детерминированную графическую игру (DG) как позиционную игру с нулевой суммой для двух человек с идеальной информацией, без случайных позиций и с конечными выплатами. В [3] эту концепцию расширили на понятие детерминированной графической многопользовательской игры (DGMP), которая может быть не с нулевой суммой, даже в случае двух игроков. Такая игра моделируется конечным ориентированным графом (орграфом) G = (V, E), вершины которого разбиты на n + 1 подмножества (n - число игроков), $V = V1 \cup ... \cup Vn \cup VT$. Вершина $v \in Vi$ интерпретируется как позиция, контролируемая игроком $i \in I = \{1, ..., n\}$, в то время как вершина $v \in VT = \{a1, ..., ap\}$ является конечной позицией (или просто терминалом, для краткости), которая не имеет исходящие ребра. Направленное ребро (v, v0) интерпретируется как (допустимый) переход из позиции v в v0. Также фиксируется начальное положение $v0 \in V \backslash VT$. Такими играми я дальше и буду оперировать.

3 Основной пример графовой игры без равновесия Нэша

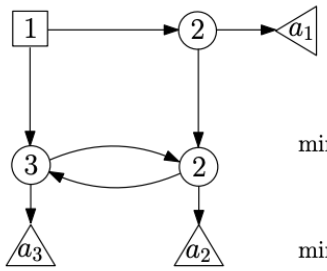
Пример приведен на 4 странице [3] Тут показаны:

1 Граф игры

2 Порядок на исходах у игроков

3 Таблица исходов в зависимости от стратегии

Конкретно эта конфигурация хороша тем, что в ней цикл - 3-ий худший исход для 2-ух игроков и самый худший для одного, а одна из целей моей работы это как раз найти игру которая удовлетворяет catch22



(a) the DGMP game structure (G, D, v_0)

$$\min\{u_1(a_2), u_1(c)\} > u_1(a_1) > u_1(a_3)$$
$$u_2(a_3) > u_2(a_1) > u_2(a_2) > u_2(c)$$
$$\min\{u_3(a_1), u_3(c)\} > u_3(a_3) > u_3(a_2)$$

(b) constraints on the payoff function u

player 2

player 1

player 3

		$\frac{(2,2)}{(2,3)}$	$\frac{(2,2)}{(2,a_2)}$	$\frac{(2,a_1)}{(2,3)}$	$\frac{(2,a_1)}{(2,a_2)}$
$(3,2)$	$(1,2)$	c	a_2	a_1	a_1
	$(1,3)$	c	a_2	c	a_2
$(3,a_3)$	$(1,2)$	a_3	a_2	a_1	a_1
	$(1,3)$	a_3	a_3	a_3	a_3

(c) the game in normal form

4 Основная схема генерации игры

Чтобы генерировать игры мне понадобился список неизоморфных графов. Сначала я написал свой скрипт который генерировал мне произвольный граф и отдельно проверял был ли до этого изоморфный ему. К сожалению такой подход оказался далеко не самым эффективным поэтому далее я просто использовал готовый список не изоморфных ориентированных графов найденный в интернете [1], далее подробнее о том как я из графов делаю игру

4.1 Графовое представление

Из графа данного на вход нужно сделать ориентированный раскрашенный в n цветов (не считая листья) (n - число игроков) граф. В таком графе листья и ориентированные циклы будут исходами игры, а остальные вершины будут вершинами в которых игрок в чей цвет раскрашена вершина делает ход. Ходом я называю выбор конкретного исходящего из вершины ребра. (реализация [2])

4.2 Каноническое представление в виде таблицы ходов

Из готового на первом шагу графа игры необходимо сделать классическую интерпретацию игры в виде n мерной таблицы размерности $d_1 \times d_2 \dots d_n$, где d_i - число стратегий

конкретного игрока, т.е. произведение исходящих степеней раскрашенных в цвет игрока вершин. Каждая ячейка таблицы кодирует партию и в ней записан ее исход. Для того чтобы заполнить такую таблицу было придумано 2 подхода:

1 Пройтись по ячейкам таблицы выбирая соответствующий подграф запускать dfs и смотреть какой соответствует исход. Предварительно найдя все циклы в исходном графе и соотнеся вершины цикла самому циклу.

2 Пройтись по графу подвесив его за начальную вершину dfs-ом и заполнить соответствующие выбранному пути по графу ячейки таблицы листьями, оставшиеся же незаполненные ячейки заполнить отдельным исходом соответствующим всем циклам

Второй подход очевидно более эффективный, я реализовал оба подхода, начав с первого и убедившись что это не тонкое место программы использовал второй подход [2]

4.3 SAT

Чтобы проверить не существование равновесий Нэша в игре я решил свести задачу к SAT-у. Для этого я буду делать КНФ с двумя видами конъюктов:

1 Первые будут показывать что из каждого исхода есть более выгодное отклонение для хотя-бы одного из игроков. Для этого возьмем n-мерные "кресты"исходов отклонений по каждой из клеток таблицы канонических представлений и сделаем конъюкт равный дизъюнкции сравнений исходов для делающего отклонение игрока. Например $((v_k(x_i) > v_k(out)) \vee \dots)$, где v_k - функция определяющая выигрыш к-того игрока в зависимости от исхода, out - исход в ячейке, x_i - одно из множества отклонений к-того игрока. Такие конъюкты показывают что в каждом исходе есть выгодное отклонение хоть для одного из игроков, следовательно ни в одном исходе нет равновесия Нэша.

2 Вторые будут показывать что у нас нет циклов в порядках функций определяющих выигрыш (например $v(out_1) > v(out_2) > v(out_3) > v(out_1)$ - это плохо) для этого сгенерируем конъюкты в которых переберем все возможные циклы из дизъюнктов из предыдущего пункта и навесим таким скобкам отрицание

Запустив SAT-solver с таким КНФ мы получим что КНФ означиваема тогда и только тогда когда не существует равновесия Нэша. А само означивание будет кодировать порядок на исходах для каждого из игроков. И из-за того что таких означиваний может быть несколько, для полноты исследования надо находить строго все означивания, не ограничиваясь одним.

4.4 Порядок исходов для каждого игрока

Порядок исходов достаточно несложно восстановить из означиваний КНФ-а, т.к. кажый элемент оного это порядок на двух элементах из возможных исходах для конкретного игрока. А т.к. в КНФ-е требуется еще и отсутствие циклов, то такими сравнениями получим строгий порядок для каждого игрока, по которым уже несложно проверить и catch22 и прочие гипотезы.

5 Различные подходы уменьшения времени работы программы

Из-за того что задача заключает в себе перебор всех небольших графов и построение по ним n мерной таблицы, а после и применение sat-solvera мы получаем NP-полную задачу с большим перебором, в следствии чего программа работает очень долго. Помимо банального распараллеливания обработки разных графов хочется найти как еще и упростить задачу каждого обрабатываемого потока. Есть несколько мест где можно улучшить работу программы:

5.1 Уменьшение кол-ва перебираемых графов

Первоначально, я пробовал просто перебирать графы с 4, 5 и 6 вершинами, но такой подход показал, что во первых графов на 6 вершинах очень много и проводить с каждым из них вышеописанный пайплайн это очень долго, а во вторых игр без равновесия Нэша среди них скорее всего нет, потому что для невырожденных игр необходимо хотя-бы 2 терминальные вершины, и получается остается всего максимум 4 вершины в которых игроки принимают решения. Обе эти проблемы я попытался решить с помощью предварительной фильтрации графов и последующим наращиванием терминальных вершин к вершинам принятия решений.

5.1.1 Какие графы можно убрать из перебора?

Во первых стоит отметить, что рассматривая графы без приделанных к ним терминалов стоит рассматривать только по крайней мере слабо связные графы. Такая фильтрация нужна потому что игры на несвязном графе эквивалентны играм образованным на компоненте связности с начальной вершиной, таких несвязных игр огромное множество и эффек-

тивнее их проверить один раз просто запуская программу на графах с меньшим кол-вом вершин. Далее убираем такие графы:

1 графы в которых существует вершина в которую нельзя попасть из начальной вершины (по той же причине что и несвязные)

2 графы в которых раскраска вершин эквивалентна графу биективно перераскрашеному

Еще не стоит забывать что после присоединения терминалов, симметричные графы повторяются из-за того что к симметричным вершинам могут симметрично быть присоединены терминалы и такие случаи тоже стоит отфильтровывать.

5.2 Убыстрение работы SAT-solver-a

Помимо большого кол-ва графов, очень много времени занимает сам процесс составления КНФ-а и его означивания, по этой причине были перебраны 2 способа составления таблицы исходов 4.2 и при формировании КНФ-а вместо всех членов брался только максимум по каждому из игроков. Для вычисления означиваний сначала я написал базовый SAT-solver, но он показывал плохую скорость и поэтому в рабочей версии я использую [6]

5.3 Распараллеливание вычислений

Даже применив все найденные мной эвристики, программа работала долго, поэтому для уменьшения времени работы я использовал мощности суперкомпьютера, на котором удаленно запускал программу в которой распределял графы поровну на ядра.

6 Результаты

Запуская программу [2] на графах с 4 и 5 вершинами, на данный момент я получил такие результаты:

1 в играх, на 4 нетерминальных вершинах, catch22 выполняется, т.е. нет конфигураций в которых цикл у двух игроков по крайней мере на 2 месте с конца

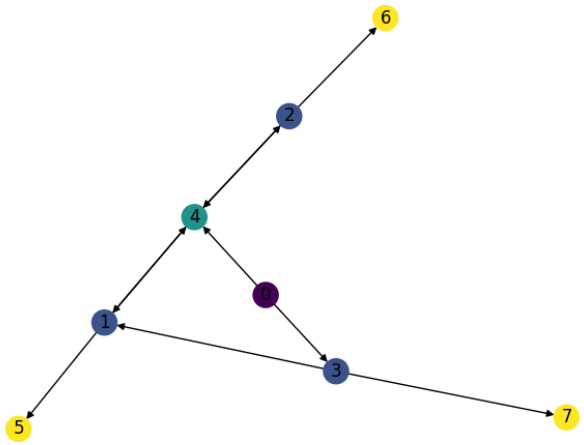
2 в играх, на 4 нетерминальных вершинах, единственная конфигурация с 0 2 2 (позициями цикла в payoff function игроков), та что была указана в [3] (она же 3) и ей эквивалентные

3 в играх, на 4 нетерминальных вершинах, нет безповторных игр без равновесий Нэша

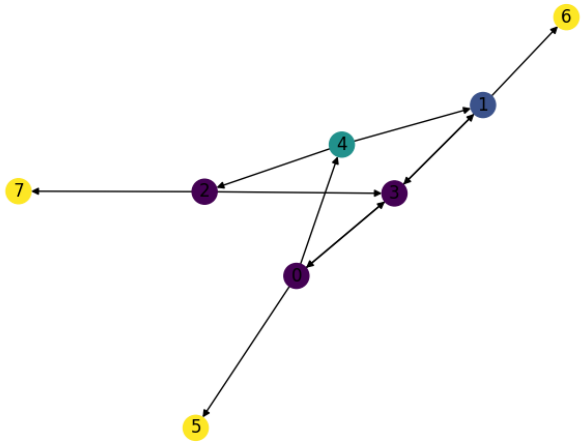
4 в играх, на 5 нетерминальных вершинах, нет безповторных игр без равновесий Нэша

5 в играх, на 5 нетерминальных вершинах с 3 игроками, catch22 выполняется, т.е. нет конфигураций в которых цикл у двух игроков по крайней мере на 2 месте с конца

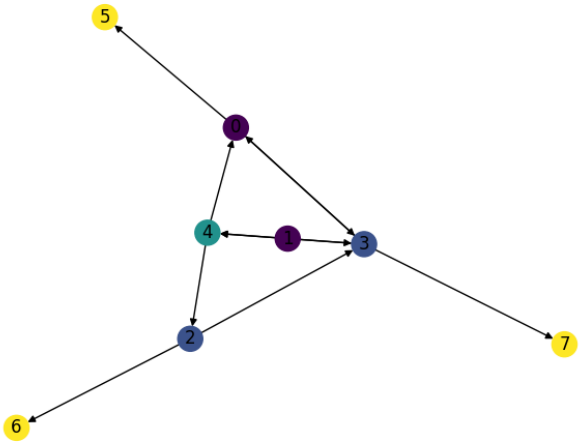
6 в играх, на 5 нетерминальных вершинах с 3 игроками, есть 215 конфигураций игр с различными графами, которые имеют конфигурацию в которой достигается 0 2 2 (некоторые из примеров таких игр ниже)



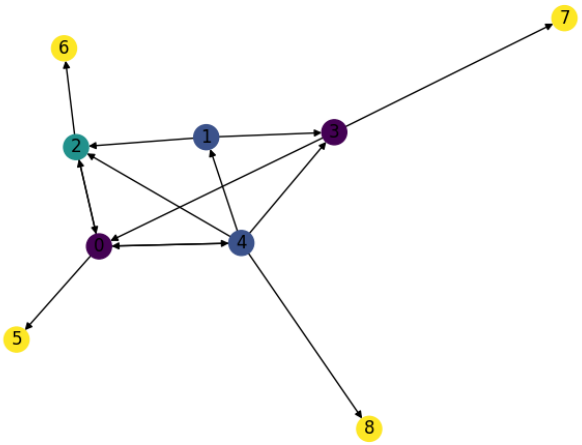
(a) Начальная вершина 0, порядки исходов: $[6, 7, c, 5][c, 5, 7, 6][5, 6, c, 7]$



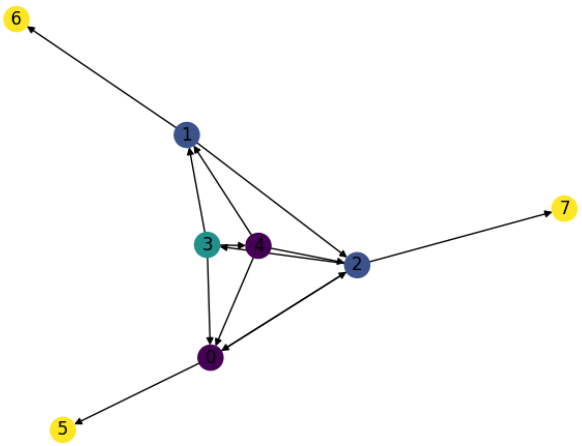
(b) Начальная вершина 4, порядки исходов: $[c, 5, 7, 6][5, 6, c, 7][6, 7, c, 5]$



(a) Начальная вершина 1, порядки исходов: $[7, 5, c, 6][c, 7, 6, 5][5, 6, c, 7]$



(b) Начальная вершина 1, порядки исходов: $[c, 8, 5, 7, 6][6, 7, c, 5, 8][5, 6, c, 7, 8]$



(c) Начальная вершина 3, порядки исходов: $[7, 5, c, 6][c, 7, 6, 5][5, 6, c, 7]$

7 Планы

Проверить гипотезы на 6 вершинах и допроверить на 5 вершинах с 4 игроками. Изучить какие еще конфигурации точно не подходят, чтобы сократить перебор и возможно смочь смотреть на графы с 7 нетерминальными вершинами.

Список литературы

[1] http://users.cecs.anu.edu.au/~bdm/data/digraphs.html.

[2] https://github.com/dimajyg/Graph-Game.

[3] Martin Milanic Endre Boros Vladimir Gurvich. "A three-person deterministic graphical game without Nash equilibria". B: *arXiv preprint, arXiv:1610.07701* (2017).

[4] Vladimir Gurvich. "On acyclicity of games with cycles". B: (2010).

[5] Vladimir Gurvich. "On Nash-solvability of finite n-person deterministic graphical games; Catch 22". B: (2021).

[6] Alexey Ignatiev, Antonio Morgado и Joao Marques-Silva. "PySAT: A Python Toolkit for Prototyping with SAT Oracles". B: *SAT*. 2018, с. 428—437. DOI: 10.1007/978-3-319-94144-8_26. URL: https://doi.org/10.1007/978-3-319-94144-8_26.

[7] J. Nash. "Equilibrium points in n-person games". B: *Proceedings of the National Academy of Sciences 36* (1950).

[8] A. R. Washburn. "Deterministic graphical games". B: *J. Math. Analysis and Applications 153* ((1990)).