Hacking Dota Underlords With Integer Programming Model

A Preprint

Alexander A. Ponomarenko, Dmitry S. Sirontkin*
Laboratory of Algorithms and Technologies for Network Analysis
National Research University Higher School of Economics
Nizhny Novgorod, Russia
aponomarenko@hse.ru, dsirotkin@hse.ru

21 июня 2020 г.

Abstract

В данной работе мы демонстрируем, как задача оптимального выбора состава команды в популярной компьютерной игре Dota UnderLords может быть сведелна к задаче целочисленного линейного программирования. Мы приводим модель и её решения. Также мы доказываем, что данная задача относится к классу NP-hard.

Keywords Целочисленное программирование · NP-трудность · NP-полнота

1 Введение

Люди любят играть в игры. Многие игры и головоломки, в которые люди играют, интересны из-за их сложности: для их решения требуется сообразительность. Часто эта трудность может быть показана математически в виде функции вычислительной сложности от размера входного набора данных. Например, было показано, что шахматы относятся к классу EXPTIME [1]; задача выбора игрока в легендарной видео-игре "Тетрис"относится классу NP-трудных [2]. Было показано, что игра-головомка "Сокобан"("кладовщик также известная для русскоговоряшего населения под названием "мудрый крот") полиномиально разрешима [4].

Особое место в теории сложности вычислений занимает класс NP-полных задач (класс NP-complete). Было показано [3], что к классу NP-полных задач относится игра "Сапёр". Задача нахождения решения требующего минимального количество передвижений фишек в обощеной версии игры 15-ки для доски размером $N \times N$ тоже относится к классу NP-complete [5]. В некотором смысле каждая NP-полная задача является загадкой, и наоборот, многие головоломки NP-полны. Для углубленого изучения темы вычислительной сложности для головоломок и игр, мы отсылаем читателя к обзору [6].

В данной работе мы уделяем внимание популярной видео-игре Dota UnlderLords. Как оказывается, данную задачу можно представить как задачу комбинаторной оптимизации, которая при фиксированном числе альяностv принадлежит классу NP-complete.

Статья организованна следующим образом. В разделе 2 повествуется в чём заключается игра Dota UnderLords. Мы приводим формулировку Dota UnderLords в виде задачи линейного целочисленного программирования разделе 3. В разделе 4 мы показываем её NP-полноту. Результаты решения модели целочисленного программирования для реальных данных мы публикуем в разделе 5. И по традиции мы подводим итоги в секции 6 "Заключение".

*Use footnote for providing further information about author (webpage, alternative address)—not for acknowledging funding agencies.

2 Описание игры Dota Underlords

По ходу игры восемь игроков составляют команду из «героев» – существ, способных сражаться друг с другом на игровой карте. У каждого из героев есть базовые параметры - здоровье, урон, скорость атаки и прочие, а также особоая способность, которая определяет его роль в игре. Каждый герой принадлежит к двум или более «альянсам» - классам, в которые входят несколько героев. так, например, герой Enchantress принадлежит одновременно к альянсу «друиды» и к альянсу «хищники». При наборе нескольких героев из одного альянса (для каждого альянса это число индивидуально) игрок получает бонус, который выражается в усилении всех героев из альянса, усилении всех своих героев или ослаблении всех героев соперника. Последнее может быть интерпретировано как относительное усиление своих героев и поэтому на протяжении работы будут рассматриваться только первые два случая. Следует отметить, что для одного альянса может быть несколько бонусов, которые открываются разным количеством героев соответствующего альянса, при этом они могут быть разного типа.

Также в ходе игры можно усилять своих героев до более высоких уровней или путём покупки внутриигровых предметов. В рамках данной работы эти аспекты учитываться не будут.

Таким образом, сила команды игрока определяется как:

1. Силой выбранных героев
2. Бонусами от альянсов в которых они состоят

3 Перевод задачи в язык линейного целочисленного программирования

3.1 Простейшая постановка задачи

Формализуем задачу следующим образом:

Будем считать, что всего у нас в есть n героев на выбор. Будем считать, что сила некоторого i-го героя опредяляется за некоторую неотрицательную величину s_i. За x_i будем обозначать принадлежность героя выбранной команде. Условимся, что когда $x_i = 1$, если i-й герой принадлежит набранной команде и $x_i = 0$ в противном случае. Тогда условие того, что в команде не более, чем m героев можно записать в виде $\sum_{i=1}^{n} x_i \leq m$. Тогда в простейшей форме данную задачу можно сформулировать следующим образом:

$$\begin{gathered} max \sum_{i=1}^{n} x_i s_i \\ \sum_{i=1}^{n} x_i \leq m \\ x_i \in \{0, 1\} \text{ – управляющая переменная} \\ n, m, s_i \text{ – костанты} \end{gathered} \tag{1}$$

В данной постановке задача решается элементарно – достаточно взять n элементов с наибольшими весами

3.2 Постановка задачи с альянсами

Как было упомянуто, в «Dota Underlords» каждый герой принадлежит к двум или более «альянсам» - классам, в которые входят несколько героев. Когда в команде присутствует несколько героев из одного альянса (для каждого альянса это число индивидуально) игрок получает бонус, который выражается в усилении всех героев из альянса, усилении всех своих героев или ослаблении всех героев соперника. Последнее может быть интерпретировано как относительное усиление своих героев и поэтому на протяжении работы будут рассматриваться только первые два случая. Следует отметить, что для одного альянса может быть несколько бонусов, которые открываются разным количеством героев соответствующего альянса, при этом они могут быть разного типа.

Данную ситуцию мы предлагаем моделировать с помощью введения 3-х индексного тензора $e_{ijk} \in \mathbb{R}$ означающего бонус герою i, за альянс j, в котором присутствуют не меннее k героев альянса j. Другими словами, e_{ijk} это k-й бонус альянса j для героя i.

С помошью тензора e_{ijk} можно поддерижвать два типа альянсов – те которые дают бонусы своим членам и те, которые дают бонусы всем героям игрока. При этом, альянсы рассмотренных видов отличются только тем, что в альянсах, дающих бонус своим членам, величина e_{ijk} равна нулю тогда и только тогда, когда i-й герой не принадлежит j-му альянсу. Заметим, что тензор e_{ijk} будет сильно разрежан, поскольку альянсы от которых бонусы идут всем гером не много. Контролировать вхождения бонуса e_{ijk} в общую силу команды предлагается с помощью управляющей бинарной переменной I_{ijk}. Так мы можем записать целевую функцию как следующую сумму $\sum_{i=1}^{n} x_i s_i + \sum_{i=1}^{n} \sum_{j=1}^{t}$. Связь переменных x_i и I_{ijk} задаётся неравенствами $\forall i, j, k : \sum_{i'=1}^{n} a_{i'j} x_{i'} - k \geq M(I_{ijk} - 1)$. Здесь надо вставить пару предложений про эти неравенства, в чём логика этой связи Они недают бинарной переменной I_{ijk} принимать значение 1, если в решение входит меньше чем k героев входящих в альянс j. Когда решение содержит героев из альянса j меньше чем k, левая часть этого неравенства отрицательная, поэтому чтобы неравентсва соблюдались правая часть должна быть ещё меньше. Такое возможно только, когда бинарная I_{ijk} равно нулю. В этом случае правая часть равна $-$, где константа заведомо большая, чем k, то есть больше, чем максимальный размер альянса q. Разумно требовать, чтобы бонус для героя i мог быть активирован ($I_{ijk} = 1$), только когда герой i входит в решение. Это задаётся неравенствами $\forall i, j, k : I_{ijk} \leq x_i$. Также мы хотим, чтобы бонус e_{ijk} был активирован, только для если герой i принадлежит альянсу j. Для этого мы в модель включили неравенства $\forall i, j, k : I_{ijk} \leq a_{ij}$.

Таким образом, после введения в модель альянсов, она выглядит следующим образом.

Целевая функция:

$$max \sum_{i=1}^{n} x_i s_i + \sum_{i=1}^{n} \sum_{j=1}^{t} \sum_{k=1}^{q} e_{ijk} I_{ijk}$$

Ограничения на входные данные

$$\forall j : \sum_{i=1}^{n} a_{ij} \leq q$$

Ограничения на управляющие переменные

$$\forall i, j, k : \sum_{i'=1}^{n} a_{i'j} x_{i'} - k \geq M(I_{ijk} - 1)$$

$$\sum_{i=1}^{n} x_i \leq m \tag{2}$$

$$\forall i, j, k : I_{ijk} \leq x_i$$

Управляющие переменные:

$I_{ijk} \in \{0, 1\}$, 1 – если для героя i, активированk-й бонус j-го альянса,

$x_i \in \{0, 1\}$, 1 – если герой i– входит в решение

Константы:

$n \in \mathbb{N}$ – число героев,

$m \in \mathbb{N}$ – максимальный размер команды

$t \in \mathbb{N}$ – общее количество альянсов

$q \in \mathbb{N}$ – максимальный размер одного альянса,

$s_i \in \mathbb{R}$— сила героя i,

$e_{ijk} \in \mathbb{R}$ – бонус для героя i, если активирован k-й бонус j-го альянса

$a_{ij} \in \{0, 1\}$, 1 – если герой i входит в альянс j

4 Доказательство NP-трудной задачи Dota Underlords с альянсами

Чтобы доказать, что задача T является NP-трудной достаточно показать, что к ней может быть сведена хотя бы одна из задач про которые известно, что она является NP-трудной.

4.1 Сведение задачи о поиска максимальго плотного подграфа к UnderLoards

Рассмотрим её частный случай - пусть все альянсы имеют размер равный двум, и сила всех героев одинакова. Рассмотрим частный случай задачи Dota Underlords со следующими ограничениями:

1. Силы всех героев одинаковы ($\forall i, j s_i = s_j$)
2. Альянсы могут давать бонусы исключительно героям в них состоящим ($\forall i, j, k a_{ij} = 0 \implies e_{ijk} = 0$)
3. Все альянсы имеют одинаковый размер, равный двум ($\forall j \sum_i a_{ij} = 2$)
4. Все альянсы дают бонус исключительно при наличии в них обоих героев ($\forall i, j e_{ij1} = 0$)
5. Бонусы ото всех альянсов равны ($\forall i, j, i', j' a_{ij} = 1, a_{i'j'} = 1 \implies e_{ij2} = e_{i'j'2}$)

Тогда данные можно представить в виде графа $G(V, E)$, где множество вершин V соответствует героям, а множество рёбер E - активным альянсам. Задача поиска оптимальной команды размера k таким образом сводится к поиску порождённого графа G на k вершинах с максимальной плотностью. Под плотностью в данной формулировке может понимается величина $\frac{G'(E)}{G'(V)}$. Действительно, при данных ограничениях общая сила команды линейно зависит от количества активных альянсов, что соответствует $G'(E)$. Т.к. k неизменно, то с ростом плотности графа G' растёт итоговая сила команды.

В работе [7] было показано, что задача поиска порождённого подграфа с максимальной плотностью и фиксированным размером в графе является NP-полной. Как было показано, она является частным случаем задачи Dota Underlords и к ней сводится. Отсюда следует её NP-сложность.

4.2 Доказательство NP-полноты - сведение UnderLords к NP-complete

Покажем, что задача Dota Underlords сводится к задаче о максимальной взвешенной по рёбрам клике. Из NP-плносты decision version задачи о взвешенной клике следует NP-полнота decision version задачи Dota Underlords. В данном сведении мы будем рассматривать задачу, в которой, максимальный размер альянса q ограничен константой некоторой константой.

Доказательство будет проводится последовательно, через серию теорем, каждая из которых описывает сведение всё менее и менее ограниченной версии задачи к задаче о максимальной взвешенной по рёбрам клике.

Теорема 1. Задача Dota Underlords без альянсов сводится к задаче о максимальной взвешенной по рёбрам клике.

Доказательство. Построим граф G со взвешенными рёбрами такой, что из решения задачи MEWC следует решение задачи DU. При этом размер задачи MEWC ограничен полиномом от размера задачи DU.

Построим множество V^1 из n вершин, соответствующее множеству героев в задаче DU. Каждой вершине поставим соответствие одного из героев из задачи DU — или, иначе говоря, назовём каждую вершину в честь одного из героев задачи DU. Пронумернуем эти вершины соответственно порядку героев v_1^1, v_2^1, ..., v_n^1

Построим дополнительно $m-1$ множеств вершин V^2, V^3 и так далее до V^m, в каждом из которых также назовём по одной вершине в часть одного из героев задачи DU. Аналогично первому множеству, пронумеруем вершины в множестве V^i как v_1^i, v_2^i, ..., v_n^i.

Обозначим семейство этих множеств за $\mathcal{F}$. Таким образом мы получим m множеств по n вершин каждое, приэтом в каждом множестве есть по одной вершине, соотвествующей каждому из героев.

Проведём рёбра следующим образом — в графе между вершинами v_a^i и $v_{a'}^{i'}$ проводится ребро, если выполняются оба следующих условия:

- Вершины v_a^i и $v_{a'}^{i'}$ соответствуют разным героям ($a \neq a'$)
- Вершины v_a^i и $v_{a'}^{i'}$ лежат в разных множествах из семейства F ($i \neq i'$)

Расммотрим все максимальные клики в данном графе. Очевидно, что в любой такой клике есть ровно по одной вершине из каждого из множеств V^i — всего m вершин. Также все эти вершины соответствуют разным героям. Таким образом каждая клике задаёт некоторую команду из героев в задаче DU. При этом стоит отметить, что каждой команде может соответствовать несколько клик.

Теперь расставим на рёбрах веса. Для этого ребру, соединяющем вершины s_a^i и $s_{a'}^{i'}$ припишем вес $\frac{s_a}{m-1} + \frac{s_{a'}}{m-1}$. Покажем, что сумма весов рёбер в клике, соответствующей некоторой команде есть в точности сила этой команды.

Действительно, в клике для каждой её вершины есть ровно $m-1$ ребро, ей инцедентное. Тогда каждое слагаемое $\frac{s_i}{m-1}$ соответствующее вершине с нижним индексом i входит в сумму ровно $m-1$ раз. Отсюда следует, что сумма всех весов рёбер в клике есть сумма всех величин s_i, соответствующих номерам вершин, образующим данную клику. □

Теорема 2. Задача Dota Underlords с альянсами размера ровно 2, сводится к задаче о максимальной взвешенной по рёбрам клике.

Доказательство. Построим граф G' со взвешенными рёбрами такой, что из решения задачи MEWC следует решение задачи DU. При этом размер задачи MEWC ограничен полиномом от размера задачи DU.

Возьмём в качестве основы для построения граф G из теоремы 1.

Построим множество W^{12} из $\binom{n}{2}$ вершин, где каждая вершина соответствует неупорядоченной паре героев. Пронумеруем эти вершины в лексикографическом порядке, соответственно порядку пар w_{12}^1, w_{13}^1, ..., w_{n-1n}^1

Построим дополнительно $\binom{m}{2} - 1$ множеств вершин W^{13}, W^{14} и так далее до W^{m-1m}, в каждом из которых также сопоставим по вершине неупорядоченной паре героев задачи DU. Аналогично первому множеству, пронумеруем вершины в множестве V^{ij} как w_1^{ij}, w_2^{ij}, ..., w_{n-1n}^{ij}.

Обозначим семейство этих множеств за $\mathcal{F}_2$. Таким образом мы получим $\binom{m}{2}$ множеств по $\binom{n}{2}$ вершин каждое, при этом в каждом множестве есть по одной вершине, соотвествующей каждой паре героев.

Проведём дополнительные рёбра следующим образом — в графе между вершинами w_{ab}^{ij} и $w_{ab}^{i'j'}$ проводится ребро, если выполняются оба следующих условия:

- Вершины w_{ab}^{ij} и $w_{ab}^{i'j'}$ соответствуют разным парам героев ($a \neq a' \vee b \neq b'$)
- Вершины w_{ab}^{ij} и $w_{ab}^{i'j'}$ лежат в разных множествах из семейства F_2 ($i \neq i' \vee j \neq j'$)

Также проведём все рёбра между всеми вершинами множеств F и F_2. Всем свежепроведённым рёбрам припишем вес 0. Очевидно, что любая максимальная клика содержит по одной вершине из каждого из множеств V и W семейств $\mathcal{F}$ и $\mathcal{F}_2$.

Припишем каждому ребру вида (v_a^k, w_{ab}^{ij}) или (v_b^k, w_{ab}^{ij}) некоторый высокий константный вес N. Данные рёбра содиняют вершину из семейства $\mathcal{F}$, соответствующую некоторому герою с вершиной из семейства F_2, соответствующей паре героев, куда этот герой входит. Покажем, что любая максимальная клика в таком графе, содержащая множество вершин из семейства $\mathcal{F}$, соответствующих некоторому набору героев, также содержит и набор вершин из семейства $\mathcal{F}_2$, соответствующий всем парам героев из этого набора.

В данной клике, рёбра, соединяющие вершины из семейств $\mathcal{F}$ и $\mathcal{F}_2$ вносят суммарный вклад в вес, равный $2\binom{n}{2}N$, т.к. в неё входит ровно $2\binom{n}{2}$ рёбер с добавленным высоким константным весом N — по два, инцедентых каждой вершине из $\mathcal{F}_2$. Отметим, что при взятии в клику из семейства $\mathcal{F}_2$ любой вершины, не соответствующей паре взятых вершин из $\mathcal{F}$ среди рёбер клики будет менее двух рёбер с добавленным высоким константным весом N. Таким образом, данная клика не будет максимальной по весу.

Таким образом, утверждение доказано. Отметим, что добавление на рёбра весов, малых по сравнению с N сохраняет истоинность утверждения. Поскольку N берётся произвольно, можно считать, что все числовые значения сил и бонусов малы по сравнению с N.

Добавим тогда к весам рёбер вида $(v_c^k, w_{a,b}^{i,j})$, соединяющих вершины из множеств $\mathcal{F}$ и $\mathcal{F}_2$ бонусы, которые альянс из пары героев под номерами a и b даёт герою под номером c. Тогда, если в выбранной команде есть герои a, b и c, то в вес данной клики будет включён этот бонус. Поскольку во все рассматриваемые клики включено одинаковое количество рёбер с добавленным весом N, то максимальной будет та клика, где максимальной будет сумма сил героев (сумма весов рёбер между вершинами семейства $\mathcal{F}$) и бонусов (веса рёбер между вершинами семейств $\mathcal{F}$ и $\mathcal{F}_2$ без учёта констант N).

Таким образом, вес клики соответствует суммарному бонусу от команды, откуда и видно сведение.

□

4.3 Всё вместе. Небольшой итог

Вариант задачи в версии "Да/Нет".

Можно вставить про то, как да/нет версия формулируется. И как с её помощью решается задача DU за логарифмическую сложность методом бинарного поиска. Тем самым оставаясь в классе np-complete

Теорема 3. Задача Dota Underloards задаваемая системой неравенств 2 принадлежит классу NP-полных задач.

Доказательство. В предыдуших двух частях мы показали, что задача UnderLords сводится в одну и в другую сторону. Таким образом DU для фиксированного числа альянсов в версии "Да/Нет"принадлежит классу NP-полных задач. таким образом задача DU NP-трудна, и при этом сам лежит в классе NP. □

5 Практическое применение для реальной задачи Dota Underlords

Мы применяем данную модель для анализа реальной задачи Dota Underlords. Отметим, что полученные результаты не стоит считать некоторой объективной оценкой качества команды героев. Причина состоит в неизбежном упрощении сил героев и влияния, которое оказывают альянсы. Каждый герой в Underlords обладает некоторой способностью, которая активируется при заполнении шкалы маны и обладает некоторым временем перезарядки. Способности и бонусы альянсов также весьма разнообразны по своему влиянию на игру - они могут наносить урон, лечить союзников, мешать врагам пользоваться своими способностями и прочее. К счастью, в игре есть система из пяти «ярусов», устроенная так, что герои внутри яруса примерно равны по силе.

В рамках упрощённой модели мы принимаем следующее.

1) Силы всех героев первого яруса равны 1, второго 2, третьего - 3, чевертого - 4, пятого - 5 2) Альянсы дают один и тот же процентный бонус всем, на кого они одинаково влияют. 3) Бонус от альянса составляет примерно 10-30 процентов от силы героя.

Детализированная таблица может быть найдена в нашем репозитории [8].

Таблица альянсов

Таблица результатов выглядит следущим образом 5:

6 Заключение

По традиции мы завершаем нашу статью разделом "заключение". В работе мы продемонстрировали к Результаты и Jupyter-тетради могут быть найдены в репозитории по адресу. Мы надеемся, что наша статья о популярной видео-игре поможет привлеч внимание молодых умов к целочисленному программированию, методам дискретной оптимизации, а также к проблеме тысячилетия P=?NP.

И важно, что мы показали что поиск такого-то оптимального (записать формальным языком)

Важно, что математическая постановка задачи задаваемая набором неравенств (2) может рассматриваться сама по себе, абстрагируясь от предметной области. И в данной работе показано, что кажущаяся

№	Герой	Сила	Альянсы
0	tusk	1	savage, warrior
1	venomancer	1	scaled, summoner
2	shadow demon	1	demon, heartless
3	drow ranger	1	heartless, hunter, vigilant
4	bloodseeker	1	blood-bound, deadeye
5	nyx assassin	1	assassin, insect
6	crystal maiden	1	human, mage
7	tiny	1	primordial, warrior
8	batrider	1	knight, troll
9	magnus	1	druid, savage
10	snapfire	1	brawny, dragon
11	arc warden	1	primordial, summoner
12	razor	1	mage, primordial
13	weaver	1	hunter, insect
14	warlock	1	blood-bound, healer, warlock
15	dazzle	2	healer, troll
16	earth spirit	2	spirit, warrior
17	storm spirit	2	mage, spirit
18	witch doctor	2	troll, warlock
19	bristleback	2	brawny, savage
20	legion commander	2	champion, human
21	queen of pain	2	assassin, demon
22	nature's prophet	2	druid, summoner
23	luna	2	knight, vigilant
24	windranger	2	hunter, vigilant
25	ogre magi	2	blood-bound, brute, mage
26	pudge	2	heartless, warrior
27	beastmaster	2	brawny, hunter
28	chaos knight	2	demon, knight
29	slardar	2	scaled, warrior
30	abaddon	3	heartless, knight
31	viper	3	assassin, dragon
32	juggernaut	3	brawny, warrior
33	ember spirit	3	assassin, spirit
34	io	3	druid, primordial
35	shadow fiend	3	demon, warlock
36	lycan	3	human, savage, summoner
37	broodmother	3	insect, warlock
38	morphling	3	mage, primordial
39	lifestealer	3	brute, heartless
40	omniknight	3	healer, human, knight
41	terrorblade	3	demon, hunter
42	shadow shaman	3	summoner, troll
43	enigma	3	primordial, void
44	treant protector	3	brute, druid
45	doom	4	brute, demon
46	disraptor	4	brawny, warlock
47	void spirit	4	spirit, void
48	mirana	4	hunter, vigilant
49	tidehunter	4	scaled, warrior
50	necrophos	4	healer, heartless, warlock
51	lone druid	4	druid, savage, summoner
52	sven	4	human, knight, scaled
53	slark	4	assassin, scaled
54	templar assassin	4	assassin, vigilant, void
55	keeper of the light	4	human, mage
56	axe	5	brawny, brute
57	faceless void	5	assassin, void
58	sand king	5	insect, savage
59	lich	5	heartless, mage
60	medusa	5	hunter, scaled
61	dragon knight	5	dragon, human, knight
62	troll warlord	5	troll, warrior

Таблица 1: Таблица силы геров и принадлежности их к альянсам.

Герой													Вклад альянса	Сила героев	сумма
broodmother	heartless 2 +0.3	human 2 +0.3	insect 2 +0.3	scaled 2 +0.6	troll 2 +0.3	warlock 2 +0.6	warlock 4 +0.6						3.0	3	6.0
disraptor	heartless 2 +0.4	human 2 +0.4	insect 2 +0.4	scaled 2 +0.8	troll 2 +0.4	warlock 2 +0.8	warlock 4 +0.8						4.0	4	8.0
dragon knight	heartless 2 +0.5	human 2 +0.5	insect 2 +0.5	knight 2 +1.0	scaled 2 +1.0	troll 2 +0.5	warlock 2 +0.5	warlock 4 +0.5					5.0	5	10.0
lich	heartless 2 +0.5	human 2 +0.5	insect 2 +0.5	scaled 2 +1.0	troll 2 +0.5	warlock 2 +0.5	warlock 4 +0.5						4.0	5	9.0
medusa	heartless 2 +0.5	human 2 +0.5	insect 2 +0.5	scaled 2 +1.0	troll 2 +0.5	warlock 2 +0.5	warlock 4 +0.5						4.0	5	9.0
necrophos	heartless 2 +0.4	human 2 +0.4	insect 2 +0.4	scaled 2 +0.8	troll 2 +0.4	warlock 2 +0.8	warlock 4 +0.8						4.0	4	8.0
sand king	heartless 2 +0.5	human 2 +0.5	insect 2 +0.5	scaled 2 +1.0	troll 2 +0.5	warlock 2 +0.5	warlock 4 +0.5						4.0	5	9.0
sven	heartless 2 +0.4	human 2 +0.4	insect 2 +0.4	knight 2 +0.8	scaled 2 +0.8	troll 2 +0.4	warlock 2 +0.4	warlock 4 +0.4					4.0	4	8.0
troll warlord	heartless 2 +0.5	human 2 +0.5	insect 2 +0.5	scaled 2 +1.0	troll 2 +1.0	warlock 2 +0.5	warlock 4 +0.5						4.5	5	9.5
witch doctor	heartless 2 +0.2	human 2 +0.2	insect 2 +0.2	scaled 2 +0.4	troll 2 +0.4	warlock 2 +0.4	warlock 4 +0.4						2.2	2	4.2

Таблица 2: Оптимальный состав команды в для игры Dota UnderLoards. Таблица результатов

на первый взгляд np-hard задача, всё-таки является np-полной. что это также вклад в исследование np-послных задач. что сама по себе задача описанная неравенствами

Список литературы

[1] Aviezri S Fraenkel and David Lichtenstein. Computing a perfect strategy for n× n chess requires time exponential in n. In International Colloquium on Automata, Languages, and Programming, pages 278–293. Springer, 1981.

[2] Ron Breukelaar, Erik D Demaine, Susan Hohenberger, Hendrik Jan Hoogeboom, Walter A Kosters, and David Liben-Nowell. Tetris is hard, even to approximate. International Journal of Computational Geometry & Applications, 14(01n02):41–68, 2004.

[3] Richard Kaye. Minesweeper is np-complete. The Mathematical Intelligencer, 22(2):9–15, 2000.

[4] Robert A Hearn and Erik D Demaine. Pspace-completeness of sliding-block puzzles and other problems through the nondeterministic constraint logic model of computation. Theoretical Computer Science, 343(1-2):72–96, 2005.

[5] Daniel Ratner and Manfred K Warmuth. Finding a shortest solution for the n× n extension of the 15-puzzle is intractable. In AAAI, pages 168–172, 1986.

[6] Diogo M Costa. Computational complexity of games and puzzles. arXiv preprint arXiv:1807.04724, 2018.

[7] Rod G Downey and Michael R Fellows. Fixed-parameter tractability and completeness ii: On completeness for w [1]. Theoretical Computer Science, 141(1-2):109–131, 1995.

[8] Dmitry Sirotkin Alexander Ponomarenko. github repository of supporting matterials for the Dota Underloars paper. `https://github.com/aponom84/UnderLords/blob/master/UnderLordsData.xlsx`,. Accessed: 2010-09-30.