Исследование алгоритмических игр на NP-полноту

Малюгин Руслан, Б05-921

11 января 2022 г.

Аннотация

В данной работе произведен разбор нескольких алгоритмических игр, исследование их на полноту, анализ решения и поиск приближенного решения.

1 Обзор NP-сложных задач

Напомним определение класса NP и NP-полных задач.

Определение 1 *Класс NP-множество задач разрешимости, решение, которых можно найти на машине Тьюринга за полиномиальное время от длины входа с использованием некоторого сертификата полиномиальной длины.*

Более формально:

Классом NP называется множество языков L для которых существует функция V(x,s) с булевыми значениями, вычислимая за полиномиальное время от длины первого аргумента, такая что:

- *Если* $x \in L, \exists\ s : V(x, s) = 1$.
- *Если* $x \notin L, \forall : V(x, s) = 1$.

Определение 2 *Пусть A и B - это два языка. Тогда A сводится по Карпу к B, если существует всюду определённая функция* $f : \{0,\ 1\}^* \rightarrow \{0,\ 1\}^*$ *, вычислимая за полиномиальное время, такая что* $x \in A \Leftrightarrow f(x) \in B$. *Обозначение* $A \leqslant_p B$.

Определение 3 *Язык B является NP-трудным, если для любого* $A \in NP$ *выполнено* $A \leqslant_p B$. *Язык B является NP-полным, если он NP-трудный и лежит в NP.*

Задачи из класса NP постоянно встречаются в различных сферах жизнедеятельности: восстановление поврежденных файлов, оптимизация маршрутов, сложные вычисления в биоинформатике. Криптография открытых ключей основывается на предположении, что $NP \neq P$. Если найдется способ решать задачи этого класса за полиномиальное время, то многие методы защиты больше не будут иметь смысла.

Рассмотрим некоторые наиболее популярные NP-полные задачи.

- Задача коммивояжера.

  Условие задачи содержит список городов и список расстояний между ними. NP-полная версия задается вопросом, есть ли маршрут, не длиннее заданного положительного числа K.

- Задача выполнимости булевых формул.

  Задача заключается в следующем: можно ли назначить всем переменным, встречающимся в формуле, значения ложь и истина так, чтобы формула стала истинной.

- Задача о рюкзаке.

  В общем виде задачу можно сформулировать так: из заданного множества предметов со свойствами «стоимость» и «вес» требуется отобрать подмножество с максимальной полной стоимостью, соблюдая при этом ограничение на суммарный вес.

- Задача о вершинном покрытии.

  Задача состоит в поиске вершинного покрытия наименьшего размера для заданного графа (этот размер называется числом вершинного покрытия графа).

NP-полные задачи считаются наиболее "сложными"задачами среди NP-задач. Для них скорее всего уже не будет найдено более быстрое точное решение.

Так же необходимо упомянуть саму гипотезу P=NP? Задача заключается в установлении равенства(или неравенства) между классами P и NP. Проблема равенства классов P и NP является одной из семи задач тысячелетия, за решение которой Математический институт Клэя назначил премию в миллион долларов США.

2 NP и NP-полные игры

Многие алгоритмические игры на самом деле являются NP-полными. При этом в большинстве случаев NP-полной задачей является поиск оптимального решения(за минимальное количество ходов) или поиск решения за k ходов. Так же в некоторых случаях рассматривается вопрос возможности решения игры при заданной начальной конфигурации. Далее рассмотрим некоторые примеры.

2.1 Пятнашки(игра в 15)

Игра в 15, пятнашки, такен — популярная головоломка, придуманная в 1878 году Ноем Чепмэном. Головоломка представляет собой набор из 15 одинаковых квадратных костяшек с нанесёнными на них числами, лежащих в квадратной коробке. Длина стороны коробки в четыре раза больше длины стороны костяшки, поэтому в коробке остаётся незаполненным одно квадратное поле. Цель игры — упорядочить костяшки по возрастанию номеров, перемещая их внутри коробки, желательно сделав как можно меньше перемещений.

NP-полной задачей в данном случае является поиск оптимального решения(за наименьшее количество ходов).

Эта задача является хорошим примером для исследования и поиском приближенных решений, а так же подбора наиболее удачных эвристических функций потому что:

- Не известно алгоритма, находящего решения для игры n*n за разумное время.
- Задача поиска решения идейно простая и довольно просто описывается несколькими правилами.
- Размер графа игры растет экспоненциально, при том, что каждое состояние описывается за O(n).
- Головоломка просто моделируется.
- Из-за наличия большого количества состояний игры она хороша для подбора и тестирования различных эвристических функций.

Доказательство NP-полноты довольно трудное, поэтому приведем основную идею, полное доказательство будет приложено в статье ниже.

Для начала формализуем игру.

Определение 4 *Доска - неорентированный конечный простой граф G(V,E).*

Правильной конфигурацией доски назовем такую конфигурацию, где в каждой ячейке , кроме одной лежит одно число, а одна ячейка пустая. То есть функция BC : V 1, 2, ... |V| - 1 определяет правильную конфигураци, если она биективна и если в ячейке нет числа, то BC(v)=0, и наоборот(BC(v) $\neq$ 0) если она не пуста.

Правильным ходом назовем такой ход, в ходе которого число из занятой ячейки передвигается в незанятую. На самом деле правильный ход - это преобразование на наборе правильных

конфигураций. Пусть BC(.) - это правильная конфигурация, а BC'(.) - конфигурация, полученная перестановкой одного числа. Тогда существует две смежные вершины u и v, такие, что BC(v)=BC'(u), BC(u)=BC'(v)=0 и для всех остальных вершин, кроме u и v: BC(w)=BC'(w).

Последовательностью ходов назовем последовательность $BC_0(.), BC_1(.), \ldots, BC_t(.)$*, такую, что* $\forall i \in \{1, \ldots, t\}$ $BC_{i+1}(.)$ *является результатом правильного хода на правильной конфигурации* $BC_i(.)$*.* $BC_0(.), BC_t(.)$ *назовем начальной и финальной конфигурацией соответственно*

Будем говорить, что игра имеет решение если для правильных конфигураций $BC_0(.), BC_t(.)$ *есть последовательность правильных ходов, где* $BC_0(.), BC_t(.)$ *являются начальной и финальной конфигурацией.*

С помощью данной модели определим задачу.

Определение 5 *Задача поиска наименьшего решения игры.*

Входные данные*. Граф* $G(V, E)$ *, правильные конфигурации* $BC(.), FC(.)$*, целое число k.*

Задача*. Существует ли последовательность из k(или короче) правильных ходов, которая переводит конфигурацию* $BC(.)$ *в* $FC(.)$*?*

Теорема 1 *Описанная выше задача - NP-полная.*

Определение 6 *Задача 3-покрытия.*

Входные данные*. Множество* $U = \{e_i\}_{i=1}^{3n}$ *и множество* $V = \{s_j\}_{j=1}^{m}$ *его трехэлементных подмножеств.*

Задача*. Существует ли* $V' \subseteq V$*, такое , что каждый элмент ; содержится ровно в одном множестве из* V'*.*

Доказательство 1 *К данной задаче можно свести задачу 3-покрытия множества, которая является NP-полной.*

Доказательство NP-полноты 3-покрытия будет приложено в статье ниже.

Рассмотрим как эта задачу можно решать на практике.

Мы можем рассмотреть полный граф игры и искать в нем кратчайший путь. Из известных методов поиска кратчайшего пути применимым является лишь A* или его модификации, ведь в графе количество вершин будет равным $O\left(n^2!\right)$ и полный обход будет слишком долгим(уже при n=4, число вершин более 2e+13). Так что необходимо использовать алгоритмы с эвристиками.

В качестве эвристики можно рассмотреть манхэттонское расстояние. Это сумма расстояний по строкам и столбцам от текущего расположения костяшки до ее правильной позиции. Допустимость следует из того, что за один ход перемещается только одна фишка, и расстояние между этой фишкой и её конечной позицией изменяется на 1.

Однако, к сожалению данное решение работает уже достаточно долго для игры 5*5, а для игры 6*6 время на решение превышает время жизни среднестатистического человека.

2.2 Сапер

Плоское или объёмное игровое поле разделено на смежные ячейки (квадраты, шестиугольники, кубы и т. п.), некоторые из которых «заминированы»; количество «заминированных» ячеек известно. Целью игры является открытие всех ячеек, не содержащих мины. В каждой из ячеек находится мина или число, которое означает количество мин в соседних 8 ячейках.

Будем считать, что нам дано начальное расположение некоторых чисел, на основании которого надо определить расположение мин и безопасных клеток. В этой версии вновь открытые безопасные клетки не будут показывать количество мин вокруг них. Очевидно, что не любые конфигурации игры имеют решение, например:

Рис. 1: Конфигурация, не имеющая решения

Определим задачу при данной игре, для этого введем следующее определение.

Определение 7 *Конфигурацией поля назовем начальное положение ячеек с числами.*

Назовем конфигурацию правильной, если существует хотя бы одна расстановка мин и безопасных полей, при которой игра имеет решение(не приводит к противоречию).

Окрестностью ячейки назовем позиции, индексы, которых отличаются от ее индексов на ± 1. *Задача о проверке конфигурации на правильность.*

***Входные данные**. Начальные ячейки с числами. Более формально - матрица N*M, где на позиции (i,j) стоит натуральное число(разумно рассматривать конфигурации, состоящие из неотрицательных чисел), если в этой ячейке оно есть, или же -1 иначе.*

***Задача**. Существует ли такая расстановка мин и безопасных ячеек, что при данная конфигурация является правильной. Более формально - можно ли заменить некоторые -1 в матрице на -2(не суть важно, главное иметь различие между пустыми клетками вначале, минами и отмеченными полями в конфигурации), так, чтобы числа в клетках, не равные -1, были в точности равны количеству клеток с -2 в их окрестности.*

Таким образом мы формализовали задачу и можем показать следующее утверждение.

Теорема 2 *Описанная выше задача - NP-полная.*

Доказательство 2 *Покажем сначала, что задача лежит в NP. В качестве сертификата предъявим расположение мин. Мин может быть не больше* n^2, *для каждой нужно проверить максимум 8 клеток, то есть сертификат полиномиален.*

Покажем, что задача является NP-трудной. Будем рассматривать поле достаточно большого размера. Тогда на нем можно будет изобразить логические вентили. Предъявим их изображение в явном виде.

Наличие мины в конкретной точке будем считать за 1, отсутствие за 0.

Предъявлять доказательство того, что эти приведенные ниже конфигурации верные и действительно являются логическими вентилями приводить не будем, однако это несложно проверить.

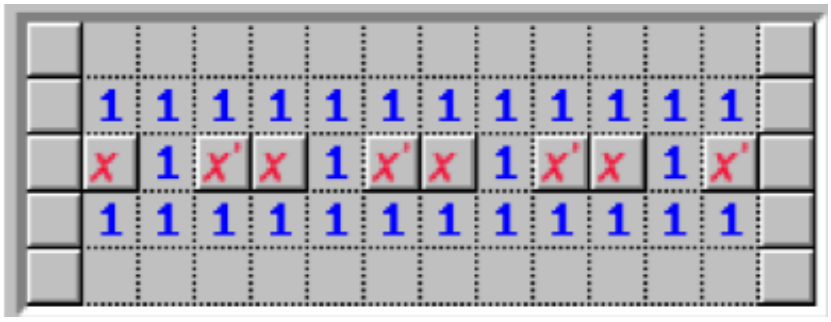

Рис. 2: Соединительный провод

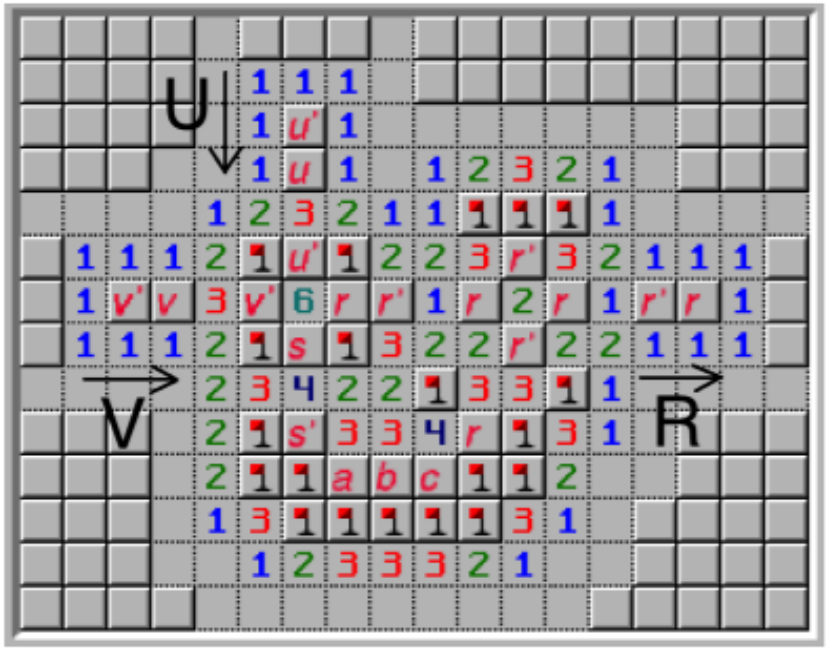

Рис. 3: OR gate

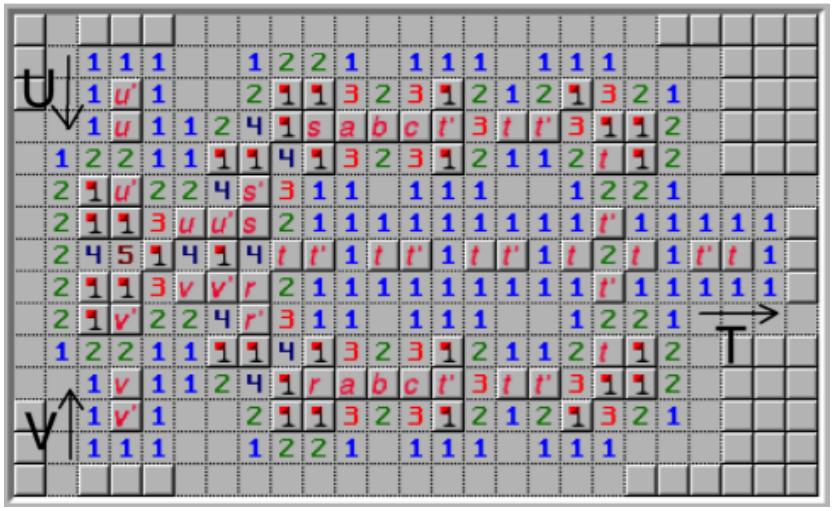

Рис. 4: AND gate

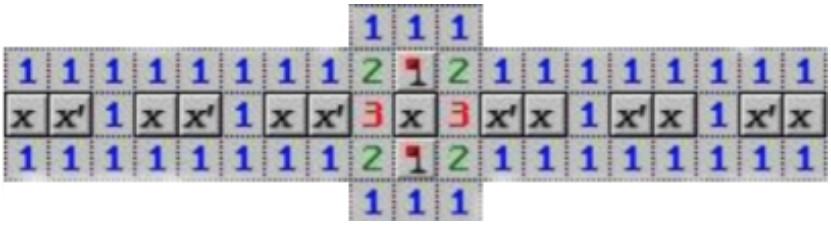

Рис. 5: NOT gate

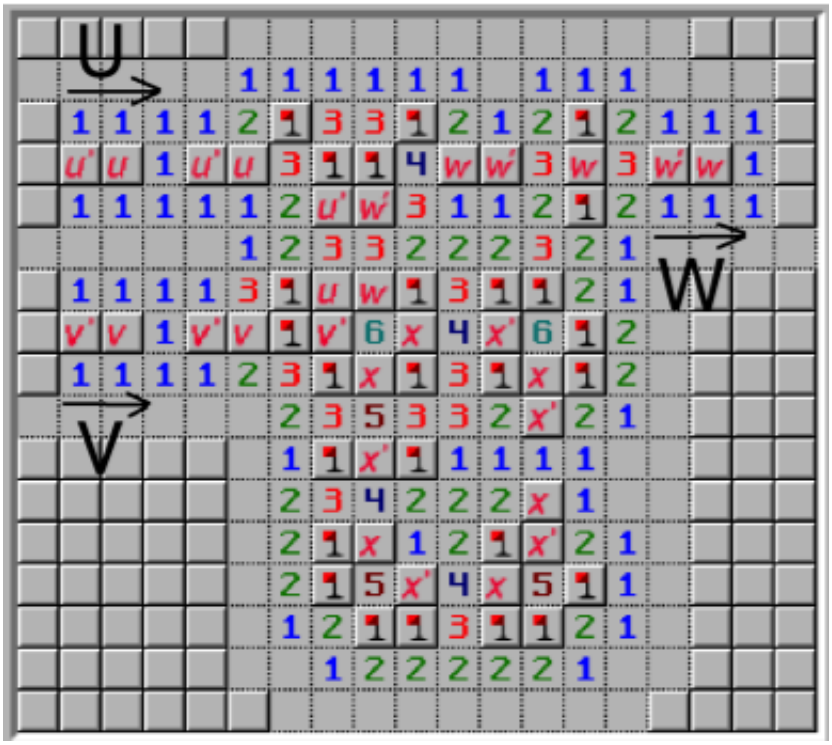

Рис. 6: XOR gate

Таким образом имея все логические вентили мы можем свести задачу о разрешимости булевых формул к саперу на достаточно большой доске.

Задача является NP-трудной и лежит в NP - значит она NP-полная.

К сожалению, так как данная задача NP-полная, то мы не можем найти достаточно быстрого решения. В некоторых случаях нам придется делать полный перебор. Но мы можем немного оптимизировать решение в особых случаях.

Возможные оптимизации:

1. Проверять поле на <однозначные> варианты расположения мин, например вокруг клеток с числом 8 ставить везде мины, клетки окруженные полями с 1 считать заминированными и т.д
2. Начинать с чистого поля и каждый раз начинать расстановку вокруг клеток с наибольшими числами.
3. Исследуем на мины только клетки, которые лежат в окрестности клеток с числами.

Аналогично алгоритм поиска решения работает очень долго уже на некоторых полях размера 5*5.

2.3 Тетрис

Случайные фигурки тетрамино падают сверху в прямоугольный стакан шириной 10 и высотой 20 клеток. В полёте игрок может поворачивать фигурку на 90° и двигать её по горизонтали. Также можно «сбрасывать» фигурку, то есть ускорять её падение, когда уже решено, куда фигурка должна упасть. Фигурка летит до тех пор, пока не наткнётся на другую фигурку либо на дно стакана. Если при этом заполнился горизонтальный ряд из 10 клеток, он пропадает и всё, что выше него, опускается на одну клетку. Дополнительно показывается фигурка, которая будет следовать после текущей — это подсказка, которая позволяет игроку планировать действия. Темп игры постепенно ускоряется. Игра заканчивается, когда новая фигурка не может поместиться в стакан. Игрок получает очки за каждый заполненный ряд, поэтому его задача — заполнять ряды, не заполняя сам стакан (по вертикали) как можно дольше, чтобы таким образом получить как можно больше очков.

Формализуем игру и поставим задачу.

Определение 8 *Игровым полем назовем клетчатый многоульник(матрицу) размера n*m, где каждая клетка свободна либо занята(0 или 1). Допустимой конфигурацией назовем такую, при которой матрица не содержит строчки из 1, и если i-ая строка состоит из 0, то все строки выше нее тоже состоят из 0.*

Фигурой называется одна из следующих комбинаций заполненных клеток(нулей). Будем называть их соответственно I, T, SQ, LG, RG, LS, RS.

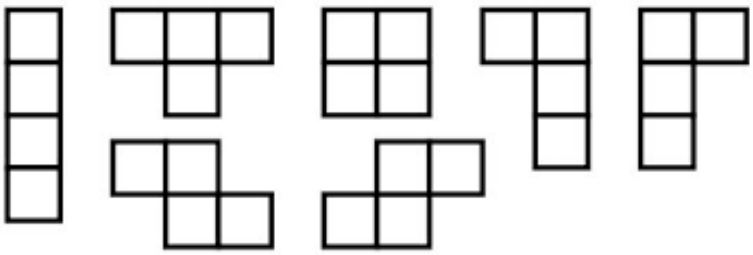

Рис. 7: Фигуры в тетрисе

Каждая фигура характеризуется 4-мя значениями.

1. *Тип*
2. *Поворот (0, 90, 180, 270)*
3. *Позиция(относительно верхней левой клетки)*
4. *Может ли фигура двигаться(фигуры за каждую итерацию движения перемещаются на строчку вниз)*

Исходной позицией фигуры называется положение ([n/2], m)

Поворот фигуры можно описать функцией $R : (P, \alpha, B) \to P'$, где P и P'-состояния фигур, α-угол -90, 90, B-игровое поле. Поворот называется допустимым, если в некоторой константной окрестности фигуры нет занятых клеток, так же если поворот допустим, то в положении P' не занимает уже занятые клетки на поле. Функция R должна выполнять следующее условие: если поворот допустим, то у фигуры меняется угол поворота на 90 градусов, а так же ее позиция, если нет, то P=P'.

Допустимые действия:

1. *Для фиксированной фигуры нет допустимых действий*
2. *Поворот против или по часовой стрелке*
3. *Сдвиг на 1 вправо или влево при наличии свободного места*

4. *Сдвиг на 1 вниз при наличии свободного места*

5. *Фиксация(если под фигурой есть занятая клетка, она фиксируется)*

Траекторией фигуры будем называть последовательность допустимых действий, которые начинаются с исходной позиции и заканчиваются фиксацией фигуры. Результатом траектории является новое поле B', определяемое следующим образом:

1. *Поле B с заполненными клетками фигуры P*

2. *Если после фиксации появились полностью заполненные строчки, мы удаляем их, и сдвигаем все поле выше на 1 вниз.*

3. *Если исходная позиция является заблокированной(нет допустимых ходов), то игра заканчивается*

Последовательностью траекторий для игры $(B_0, P_1, ..., P_k)$ *назовем последовательность* $(T_1, ..., T_k)$, *такая, что каждая траектория приводит к появлению нового поля* Bi, *а так же игра не завершается на номере хода* $i<k$, *но завершается на* k.

Решать будем следующую задачу:

***Входные данные**. Начальное положение поля B и набор фигур.*

***Задача**. Существует ли такая последовательность траекторий, что по завершению игры с поля будет удалено хотя бы s строчек?*

Докажем следующую теорему:

Теорема 3 *Задача описанная выше, является NP-полной.*

Доказательство 3 *Сертификатом является искомая последовательность. На каждом шаге итерации мы проверяем константное число клеток, а при исключении циклов размер итерации есть O(n). Таким образом сертификат полиномиален.*

Для доказательства NP- трудности сведем следующую задачу к ней.

***Определение 9** 3-partition:*

***Входные данные**. Последовательность чисел* $A = \{a_i\}_{i=1}^{3n}$ *и неотрицательное число* T, *такое, что* $\forall 1 \leqslant i \leqslant 3n : \ T/4 < a_i < T/2, \ \sum_{l=1}^{3n} a_l = nT$.

***Задача**. Можно ли разбить A на n непересекающихся множеств* $\{A_j\}_{j=1}^{n}$, *таких, что* $\forall 1 \leqslant k \leqslant n : \ \sum_{a_i \in A_k} a_i = T$.

***Теорема 4** 3-partition - NP-полный.*

***Доказательство 4** Доказательство этой теоремы весьма сложно и является частным случаем задачи partition.*

Будем считать, что partition имеет решение, если для игры, построенной по ней, можно полностью очистить поле.

Для сводимости воспользуемся следующей конструкцией.

Опишем ее: Поле имеет 6T+22+(3n+O(1)) строк и 6n+3 столбцов. Каждое a_i *будет представлено* a_i*+1 блоками размером 6×6; так как сумма элементов* $A_j = \{a_i, a_j, a_k\}$ *равна T, получаем 6(T+3)=6T+18. В дополнение к этим 6T+18 рядам, внизу находятся четыре ряда, обеспечивающие корректное положение блокам. Это поле представляет из себя n контейнеров, которые соответствуют множествам из задачи 3-partition.*

Опишем контейнер подробнее:

1. *В первом и втором столбцах пусты все клетки, кроме нижних четырех.*

2. *Третий столбец не имеет занятых клеток.*

3. *В четвертом и пятом столбцах пусты только клетки в тех строках, чей номер по модулю 6 равен 5.*

4. Шестой столбец не имеет пустых клеток.

Последняя часть - стопор, нужный для того, чтобы мы не очищали поле раньше того, как заполним все контейнеры. Опишем его:

1. В первом столбце заполнены все клетки, кроме двух верхних.

2. Во втором столбце заполнены все клетки, кроме верхней.

3. В третьей колонке пусты все клетки, кроме второй сверху.

Фигуры. Все фигуры делятся на блоки, котоорые соответствуют заполнению контейнеров. Для каждого числа a_i имеем следующую последовательность блоков:

1. Инициатор — последовательность (I,LG,SQ)

2. Наполнитель — последовательность (LG,LS,LG,LG,SQ), повторенная a_i раз

3. Завершитель— последовательность (SQ,SQ)

После частей, соответствующих a_3n, идет завершающая последовательность:

1. n фигур I подряд.

2. Одна фигура RG.

3. 3T/2+5 фигур I подряд.

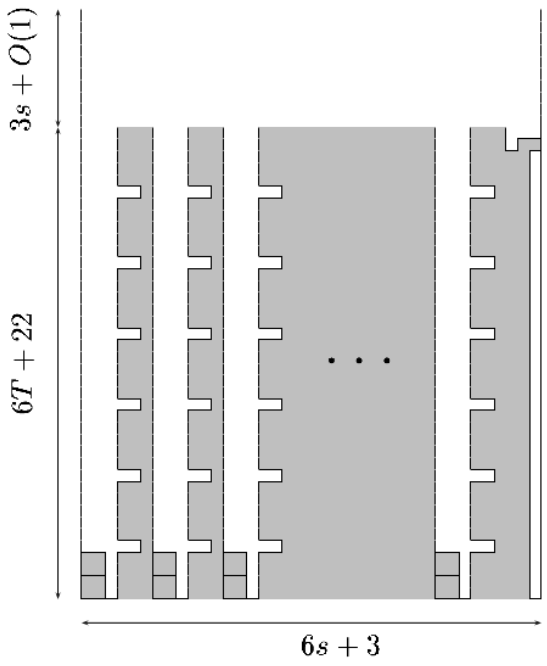

Рис. 8: Начальное расположение

Таким образом мы свели задачу partition к задаче о тетрисе, значит задача о тетрисе тоже NP-полная.

При реализации алгоритма мы можем воспользоваться техниками, полученными в прошлых задачах, а именно составить граф состояний игры и искать в нем путь. Либо мы можем для каждой фигуры разбирать все возможные ее положения, при этом запоминая сколько слоев мы уже очистили.

2.4 Краткие пояснения к некоторым другим играм

В этом разделе будут упомянуты некоторые другие игры с небольшим описанием и идеей сведения.

- Судоку. Игровое поле состоит из квадрата размером $N^2 * N^2$, разделенного на меньшие квадраты со стороной N клеток. Таким образом, всего игровое поле насчитывает N^4 клеток. В некоторых из них уже в начале игры стоят числа от 1 до N^2.

  Задача состоит в том, чтобы заполнить свободные клетки числами от 1 до N^2 так, чтобы в каждой строке, в каждом столбце и в каждом малом квадрате N*N каждое число встречалось бы ровно один раз.

  К данной задаче сводится задача о латинских квадратах, к которой в свою очередь сводится SAT или специфические задачи на графах.

- Морской бой.

  В морском бое поле представляет из себя клетчатый квадрат 10*10, на котором скрыты корабли. В состав кораблей входит один линкор длиной четыре квадрата, два крейсера длиной три квадрата, три эсминца длиной два квадрата и четыре подводные лодки размером в один квадрат. Каждый корабль занимает несколько смежных квадратов сетки, расположенных горизонтально или вертикально. Корабли располагаются так, чтобы ни один корабль не касался другого корабля даже по диагонали.

  Цель головоломки - выяснить, где находятся корабли. Сетка может начинаться с подсказок в виде уже решенных квадратов, показывающих подводную лодку, концевую часть корабля, среднюю часть корабля или воду. Рядом с каждой строкой и столбцом есть число, указывающее количество клеток, занятых частями корабля в этой строке или столбце соответственно.

  К данной задаче сводится SAT

3 Заключение

В данной работе был дан обзор на NP-полные задачи, их связь между собой. Были разобраны популярные игры, доказана их NP-полнота. Было замечено, что NP-полнота встречается не только в искусственных математических задачах, но и в обычных головоломках. На примере головоломок были показаны приложения теории сложности вычислений к реальным проблемам.

Список литературы

[1] [**https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.297.9053&rep=rep1&type=pdf**]

[2] [**https://github.com/hardenchant/15_solver/blob/master/solver15.py**]

[3] [**http://ru.discrete-mathematics.org/fall2016/3/complexity/compl-book.pdf**]

[4] [**https://arxiv.org/pdf/cs/0210020.pdf**]

[5] [**http://web.math.ucsb.edu/~padraic/ucsb_2014_15/ccs_problem_solving_w2015/NP3.pdf**]

[6] [**https://en.wikipedia.org/wiki/List_of_NP-complete_problems##Games_and_puzzles**]