МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное образовательное
учреждение высшего образования
«Ярославский государственный университет имени П. Г. Демидова»

Кафедра информатики

Сдано на кафедру
«_____»_____ 2025 г.
Заведующий кафедрой,
д. ф.-м. н., профессор

_____ В. А. Бондаренко

Выпускная квалификационная работа

Алгоритмы решения головоломки "Мосты"

по направлению
01.03.02 Прикладная математика и информатика

Научный руководитель
д.ф.-м.н., заведующий
кафедрой

_____ А. В. Николаев
«_____»_____ 2025 г.

Студент группы ИВТ-42БО

_____ А. С. Ипатов
«_____»_____ 2025 г.

Ярославль, 2025

Реферат

Объем 15 с., 3 гл., 0 рис., 0 табл., 9 источников, 0 прил.

Ключевые слова: **Головоломка, мосты, задача целочисленного линейного программирования (ЦЛП), алгоритм, оптимизация**

Объектом исследования является головоломка «Мосты».

Цель работы – разработка и реализация алгоритмов решения головоломки «Мосты» на основе методов целочисленного линейного программирования (ЦЛП), а также сравнительный анализ эффективности различных подходов. В ходе исследования были разработаны и протестированы три алгоритма.

В результате реализованы три алгоритма для решения головоломки «Мосты», использующие методы ЦЛП, проведено их сравнение и анализ.

Содержание

Введение

Hashiwokakero, также известная как «Мосты», – это японская логическая головоломка, в которой игроку нужно соединить острова мостами так, чтобы получилась связная сеть, соблюдая определенные правила.

Игра стала популярной в Японии в конце 1980-х и начале 1990-х годов благодаря Нобухиро Канаю, автору других известных головоломок, таких как Sudoku и Kakuro. Hashiwokakero была опубликована японской компанией Nikoli. Она стала одной из их самых популярных игр, вышедших под их лейблом.

В данной работе мы сосредоточимся на решении Hashiwokakero с применением методов целочисленного линейного программирования (ЦЛП). Целью является разработка и анализ нескольких алгоритмов на основе ЦЛП для эффективного решения головоломки. Для достижения этой цели были поставлены следующие задачи:

- Разработка трех различных алгоритмов решения головоломки Hashiwokakero с использованием ЦЛП.
- Реализация разработанных алгоритмов.
- Проведение сравнительного анализа эффективности алгоритмов на различных по сложности игровых полях.

1. Описание головоломки «Мосты»

1.1. Правила игры

Цель игры — соединить все острова, проведя серию мостов между ними, согласно подсказкам. В хашивокакеро играют на прямоугольной сетке нестандартного размера. Каждую ячейку можно рассматривать как остров, а число на каждом острове указывает количество мостов, которые должны быть проведены от этого острова. Числа варьируются от 1 до 8 включительно, а остальные ячейки сетки пусты. Мосты соединяют острова, образуя единую систему (рис. 1). Они должны соответствовать следующим критериям:

1) все острова должны быть соединены;
2) мосты не могут пересекаться;
3) мосты устанавливаются только по вертикали и горизонтали, никогда по диагонали;
4) не более двух мостов соединяют пару островов;
5) количество мостов, соединенных с каждым островом, должно соответствовать количеству мостов на этом острове;
6) мосты должны соединять острова в единую связную группу.

Решение головоломки требует особого планирования. Рекомендуется начинать с более простых задач и обращать внимание на острова с числами 1, 2, 7 и 8, чтобы найти отправную точку.

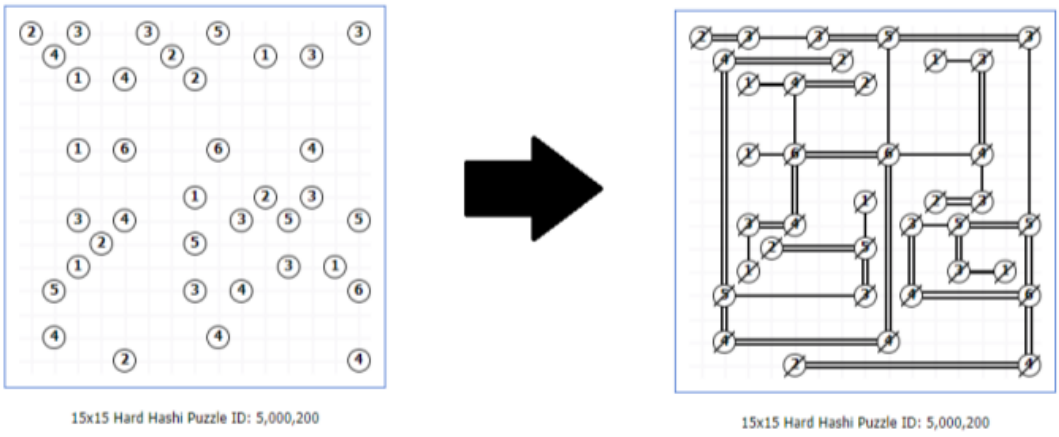

Рис. 1 — Пример решения головоломки

1.2. Представление в виде графа

Головоломку «Мосты» можно рассматривать как задачу о построении графа. Острова соответствуют вершинам графа, а мосты — ребрам. Цель состоит в том, чтобы построить связный подграф, удовлетворяющий ограничениям на степени вершин (количество мостов на острове) и множеству ребер (не более двух мостов между островами). Этот подход позволяет нам использовать мощный аппарат теории графов для анализа и решения головоломки.

Для того, чтобы представить головоломку в виде модели целочисленного линейного программирования (ЦЛП) мы вводим граф $G = (V, E)$, где:

* V – множество вершин, соответствующих островам.

* E – множество всех потенциальных мостов.

Обозначим через d_i – требуемое число мостов для острова i (т.е. степень вершины i), а через δ_i – множество всех соседей острова i (все потенциально смежные вершины графа). Кроме того, пусть Δ – множество всех пар пересекающихся мостов/рёбер $\{(i, j), (k, l)\} \in E$.

Каждому ребру $e = (i, j) \in E$ мы сопоставим две бинарные переменные:

$$x_{i,j} = \begin{cases} 1, & \text{если острова } i \text{ и } j \text{ соединены мостом,} \\ 0, & \text{в противном случае;} \end{cases}$$

$$y_{i,j} = \begin{cases} 1, & \text{если острова } i \text{ и } j \text{ соединены двойным мостом,} \\ 0, & \text{в противном случае.} \end{cases}$$

Выбор бинарных переменных обусловлен удобством представления ограничений в виде линейных неравенств, необходимых для построения модели ЦЛП. Переменная $x_{i,j}$ равна 1, если между островами i и j существует один мост, и 0 в противном случае. Аналогично, переменная $y_{i,j}$ равна 1, если между островами i и j существует двойной мост, и 0 – если его нет. Сумма $x_{i,j} + y_{i,j}$ представляет собой общее количество мостов между островами i и j.

2. Алгоритмы решения на основе ЦЛП

Представление головоломки в виде графа, как мы только что показали, позволяет нам сформулировать задачу поиска решения в виде математической модели. Одним из мощных инструментов для решения таких задач являются методы целочисленного линейного программирования (ЦЛП). ЦЛП предоставляет фреймворк, позволяющий эффективно находить оптимальные решения для широкого класса задач, включая NP-полные задачи, такие как "Мосты".

Несмотря на то, что головоломка "Мосты"относится к классу NP-полных задач, для её решения существует множество различных подходов. Однако, в данной работе мы сосредоточимся на использовании методов ЦЛП. Модели ЦЛП позволяют эффективно находить оптимальные решения для задач небольшого и среднего размера, а также служат основой для разработки более сложных и эффективных алгоритмов. В следующих разделах мы рассмотрим три различных подхода к решению головоломки "Мосты"с использованием ЦЛП:

- Итеративный ЦЛП с добавлением ограничений разрезов.
- Потоковая модель ЦЛП.
- Модифицированный итеративный ЦЛП с локальным поиском.

Каждый из этих подходов имеет свои преимущества и недостатки, и их сравнительный анализ будет представлен далее в работе.

2.1. Итеративный ЦЛП с добавлением ограничений разрезов

Модель строится на основе следующих ограничений:

1. **Ограничения степени вершин:**

$$\sum_{j \in V \setminus \{i\}} x_{i,j} + \sum_{j \in V \setminus \{i\}} y_{i,j} = d_i, \quad \forall i \in V, \tag{1}$$

   где d_i – требуемая степень острова i. Ограничение гарантирует, что каждый остров будет соединен мостами в соответствии со своей требуемой степенью;
2. **Ограничения кратных рёбер:**

$$y_{i,j} \leqslant x_{i,j}, \quad \forall (i,j) \in E. \tag{2}$$

Ограничение гарантирует, что второй мост между двумя островами может быть добавлен только в том случае, если уже есть первый мост;

3. **Ограничения пересечений:**

$$x_{i,j} + x_{k,l} \leqslant 1, \quad \forall\{(i,j),(k,l)\} \in \Delta, \tag{3}$$

где Δ – множество пар пересекающихся рёбер. Ограничение запрещает пересечение мостов;

4. **Ограничение слабой связности:**

$$\sum_{(i,j)\in E} x_{i,j} \geqslant |V| - 1.$$

Ограничение обеспечивает минимальное количество мостов для связности;

5. **Ограничение разрезов (для связности):**

$$\sum_{i\in S, j\in V\setminus S} x_{i,j} \geqslant 1, \quad \forall S \subset V.$$

Ограничение гарантирует, что между любым подмножеством островов S и остальными островами должна быть хотя бы одна связь.

Предложенный алгоритм итеративно решает модель ЦЛП, добавляя ограничения разрезов только по мере необходимости для обеспечения связности решения. Это позволяет избежать экспоненциального роста размера модели (из-за $O(2^{|V|})$ возможных ограничений разрезов) и значительно снижает вычислительную сложность.

Algorithm 1 Итеративный ЦЛП алгоритм

procedure IterativeILP($G = (V, E)$)
    Построить частичную модель ЦЛП с ограничениями (1)-(4),(6)
    repeat
        Найти решение модели ЦЛП с помощью OR-Tools (SCIP)
        if найденное решение связно **then**
            return головоломка решена
        end if
        Для каждой компоненты связности $S \subset V$ добавить ограничение (5) в модель
    until система не имеет целочисленных решений
    return головоломка не имеет решения
end procedure

2.2. Потоковая модель ЦЛП

Вторая модель целочисленного линейного программирования (ЦЛП), которую мы рассмотрим для решения головоломки "Мосты представляет собой потоковую модель. Эта модель позволяет избежать экспоненциального количества ограничений разрезов, используемых в итеративном алгоритме, заменяя их ограничениями, основанными на концепции потока в графе.

Для каждого ребра $(i, j) \in E$ в графе G мы вводим две новые неотрицательные переменные: $f_{i,j}$ и $f_{j,i}$. Эти переменные представляют собой поток по ребру в направлении от острова i к острову j, и наоборот. Выбирается некоторый остров $v_t \in V$ в качестве стока (пункта назначения потока), а все остальные острова $V \setminus \{v_t\}$ выступают в качестве источников потока, каждый с единичным объемом. Идея заключается в том, что если граф связный, то мы сможем "перегнать"единичный поток от каждого острова-источника к острову-стоку.

Таким образом, ограничения разрезов заменяются следующей системой ограничений:

1. **Ограничение стока:**

$$\sum_{i \in V \setminus \{v_t\}} f_{i,v_t} = |V| - 1.$$

   Это ограничение гарантирует, что в остров-сток v_t поступит суммарный поток, равный количеству островов-источников.
2. **Ограничение сохранения потока:**

$$\sum_{j \in V \setminus \{i\}} f_{i,j} = \sum_{j \in V \setminus \{i\}} f_{j,i} + 1, \quad \forall i \in V \setminus \{v_t\}.$$

   Для каждой вершины, отличной от стока, это ограничение гарантирует, что исходящий поток равен входящему потоку плюс единица (поток, "произведенный"в этой вершине).
3. **Ограничение пропускной способности:**

$$f_{i,j} + f_{j,i} \leqslant (|V| - 1)x_{i,j}, \quad \forall (i, j) \in E.$$

   Это ограничение устанавливает верхнюю границу на поток, который может пройти по ребру. Поток по ребру может быть только в том случае, если существует мост между островами i и j.
4. **Ограничение неотрицательности потока:**

$$f_{i,j}, f_{j,i} \geqslant 0, \quad \forall (i, j) \in E.$$

Это ограничение гарантирует, что поток не может быть отрицательным.

Дополнительно к этим ограничениям, необходимо добавить ограничения степени вершин (1), кратных рёбер (2) и пересечений (3), как и в предыдущей модели.

В отличие от итеративного алгоритма, потоковая модель позволяет решить задачу за одну итерацию, так как все ограничения для обеспечения связности включаются в модель изначально. Однако, потоковая модель может потребовать больше вычислительных ресурсов из-за большего количества переменных и ограничений. Выбор подходящего острова в качестве стока также может повлиять на производительность.

Algorithm 2 Потоковый ЦЛП алгоритм

```
1: procedure FlowBasedILP(G = (V, E))
2:     Выбрать остров v_t ∈ V в качестве стока
3:     Построить модель ЦЛП с ограничениями: (1)-(4),(6) и (7)-(10)
4:     Найти решение модели ЦЛП с помощью OR-Tools (SCIP)
5:     if найдено решение then
6:         return головоломка решена
7:     else
8:         return головоломка не имеет решения
9:     end if
10: end procedure
```

2.3. Модифицированный итеративный ЦЛП с локальным поиском

Этот подход сочетает в себе итеративное решение ЦЛП с добавлением ограничений разрезов, как в первом алгоритме, и эвристику локального поиска для улучшения сходимости и получения связного решения. Идея состоит в том, чтобы после каждой итерации ЦЛП, если решение не является связным, применить локальный поиск для "починки"решения, прежде чем добавлять новые ограничения разрезов.

Процесс локального поиска состоит в следующем:

1. **Поиск компонент связности:** Находим все компоненты связности в текущем (несвязном) решении.
2. **Выбор моста для соединения:** Выбираем остров из одной компоненты связности и пытаемся соединить его мостом с островом из другой компоненты. Выбор может быть случайным или основанным на эвристиках (например, соединять наименьшие компоненты).
3. **Фиксация моста:** Добавляем выбранный мост в решение (т.е., фиксируем соответствующую переменную $x_{i,j}$).**Разрешение конфликтов:**$(,).()$.

4. **Повторение:** Шаги 2-4 повторяются до тех пор, пока не будет получено связное решение или пока не будет достигнут предел числа итераций локального поиска.

Если после локального поиска решение все еще не является связным, добавляются ограничения разрезов, и процесс повторяется.

Эффективность этого подхода зависит от качества эвристики локального поиска и от выбора параметров (например, максимального числа итераций локального поиска). Преимущество этого метода состоит в том, что он может быстрее сходиться к решению, чем простой итеративный алгоритм, особенно для сложных головоломок.

А теперь формальное описание алгоритма:

Algorithm 3 Модифицированный итеративный ЦЛП алгоритм с локальным поиском

procedure ModifiedIterativeILP($G = (V, E)$)
    Построить частичную модель ЦЛП с ограничениями (степени, отсутствия пересечений, слабой связности)
    repeat
        Найти решение модели ЦЛП (например, с помощью OR-Tools)
        if найдено решение связно **then**
            return решение
        end if
        $G \leftarrow$ LocalSearch(G)
        if G связен **then**
            return решение
        end if
        for каждой компоненты связности $S \subset V$ **do**
            Добавить ограничение разрезов для S в модель
        end for
    until система не имеет целочисленных решений
    return "решение не найдено"
end procedure

Algorithm 4 Локальный поиск

procedure LocalSearch($G = (V, E)$)
    Находим компоненты связности в графе G и делаем случайную перестановку островов V
    for каждого острова $v \in V$ в порядке перестановки **do**
        brokenIslands $\leftarrow \varnothing$
        fixedBridges $\leftarrow \varnothing$
        if можно построить мост, соединяющий v с островом u в другой компоненте связности **then**
            $E \leftarrow E \cup (u,v)$
            fixedBridges $\leftarrow (u, v)$
            brokenIslands $\leftarrow \{u, v\}$
        end if
        while brokenIslands не пуст **do**
            Выбираем сломанный остров w из brokenIslands
            if инцидентных мостов w больше ограничения **then**
                Исключаем случайный инцидентный w мост $\neq$ fixedBridges
            else
                Добавляем, если возможно, случайный инцидентный к w мост $\neq$ fixedBridges
            end if
            Обновляем списки fixedBridges и brokenIslands
        end while
    end for
    return G
end procedure

3. Реализация и тестирование

3.1. Описание реализации

3.2. Результаты тестирования и сравнение алгоритмов

Заключение

Список литературы

[1] Cook S. A. The complexity of theorem-proving procedures // Proceedings of the third annual ACM symposium on Theory of computing. — 1971. — С. 151–158.

[2] Пупырев С. Н., Тихонов А. В. Визуализация динамических графов для анализа сложных сетей // Модел. и анализ информ. систем. — 2010. — Т. 17, № 1. — С. 117–135.

[3] Кузьмин И. Г. Некоторые проблемы государственных финансов в современной России // Российские предприятия в системе рыночных отношений, материалы научно-практич. конф. Ярославль, 17–18 окт. 2000 г. / отв. ред. Л.Б. Парфенова. — 2000. — С. 86–90.

[4] Аппаратура радиоэлектронная бытовая. Входные и выходные параметры и типы соединений. Технические требования. Введ. 2002-01-01. М // ГОСТ Р 517721-2001. — 2001. — С. 27.

[5] TeX в ЯрГУ. — Режим доступа: `http://www.tex.uniyar.ac.ru` (дата обращения: 20.05.2017).

[6] The Not So Short Introduction to LaTeX 2_ε. — Режим доступа: `https://tobi.oetiker.ch/lshort/lshort.pdf` (дата обращения: 01.06.2017).

[7] Котельников И. А., Чеботарёв П. З. LaTeX 2_ε по-русски. — Новосибирск : Сибирский хронограф, 2004. — 496 с.

[8] Tantau T. PGF— Create PostScript and PDF graphics in TeX. — Режим доступа: `https://www.ctan.org/pkg/pgf` (дата обращения: 17.05.2017).

[9] Кирютенко Ю. А. TikZ & PGF. Создание графики в LaTeX 2_ε-документах. — Ростов-на-Дону, 2014. — 277 с. — Режим доступа: `https://open-edu.sfedu.ru/files/pgf-ru-all-method.pdf`.