Задача о назначениях

Виктор Васильевич Лепин

План лекции

1. Постановка задачи

Постановка задачи

Для n работников и работ, дана матрица $n \times n$, задающая стоимость выполнения каждой работы каждым работником. Найти минимальную стоимость выполнения работ, такую что каждый работник выполняет ровно одну работу, а каждую работу выполняет ровно один работник. Т.е. произвести назначение (assignment) работника на работу.

Постановка задачи

Для n работников и работ, дана матрица $n \times n$, задающая стоимость выполнения каждой работы каждым работником. Найти минимальную стоимость выполнения работ, такую что каждый работник выполняет ровно одну работу, а каждую работу выполняет ровно один работник. Т.е. произвести назначение (assignment) работника на работу.

- Назначение это биекция ϕ между двумя конечными множествами из n элементов.

Постановка задачи

Для n работников и работ, дана матрица $n \times n$, задающая стоимость выполнения каждой работы каждым работником. Найти минимальную стоимость выполнения работ, такую что каждый работник выполняет ровно одну работу, а каждую работу выполняет ровно один работник. Т.е. произвести назначение (assignment) работника на работу.

- Назначение это биекция ϕ между двумя конечными множествами из n элементов.
- В оптимизационной задаче нужно найти наилудшее назначение, т.е. нам нужно оптимизировать некоторую целевую функцию, которая зависит от назначения ϕ.

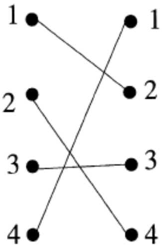
Способы представления назначения

Назначения могут быть представлены разными способами.

- Биективное отображение между двумя конечными множествами V и W может быть представлено прямым способом посредством совершенного паросочетания в двудольном графе $G = (V, W; E)$, где множества вершин V и W имеет n вершин. Ребро $(i, j) \in E$ является ребром совершенного паросочетания тогда и только тогда, когда $j = \phi(i)$, см. рис. 1.

$$\varphi = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 2 & 4 & 3 & 1 \end{pmatrix}$$

$$X_\varphi = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix}$$



Способы представления назначения

- Идентифицируя множества V и W, мы получаем представление назначения перестановкой.

Способы представления назначения

- Идентифицируя множества V и W, мы получаем представление назначения перестановкой.
- Каждая перестановка ϕ множества $N = \{1, \dots, n\}$ уникальным образом соответствует матрице перестановок $X_\phi = (x_{ij})$ с $x_{ij} = 1$ для $j = \phi(i)$ и $x_{ij} = 0$ для $j \neq \phi(i)$.
  Эту матрицу X_ϕ можно рассматривать как матрицу смежности двудольного графа G, представляющего совершенное паросочетание, см. рис.1.

Матрица весов

Определение

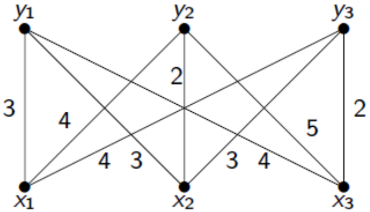
Пусть $(K_{n,n}, w)$ — взвешенный полный двудольный граф, $w(x_i, y_j) = w_{ij}$ для всех $i, j = 1, 2, \ldots, n$. Квадратная матрица $W = (w_{ij})$ порядка n называется матрицей весов этого графа.

Матрица весов

Определение

Пусть $(K_{n,n}, w)$ — взвешенный полный двудольный граф, $w(x_i, y_j) = w_{ij}$ для всех $i, j = 1, 2, \ldots, n$. Квадратная матрица $W = (w_{ij})$ порядка n называется матрицей весов этого графа.

Так, матрица на рис. справа задает взвешенный граф $(K_{3,3}, w)$ на рис. слева.



$$W = \begin{pmatrix} 3 & 4 & 4 \\ 3 & 2 & 3 \\ 4 & 5 & 2 \end{pmatrix}$$

Задача о назначениях в матричной форме

- Предположим, что 3 рабочим $I = \{I_1, I_2, I_3\}$ доступны 3 рабочих места $J = \{J_1, J_2, J_3\}$, и дана рейтинговая матрица R положительных целых чисел, где r_{ij} представляет рейтинг человека i для работы j.

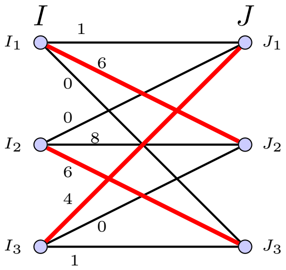
$$R = \begin{bmatrix} 1 & 6 & 0 \\ 0 & 8 & 6 \\ 4 & 0 & 1 \end{bmatrix}$$

- Вопрос: как распределять задания, чтобы получить максимальную сумму оценок (когда одному человеку назначается ровно одна работа)?
- В оригинальной статье Куна Задача о назначениях описывается как максимизация рейтингов. Альтернативная и эквивалентная постановка задачи — минимизация затрат. Например, мы можем установить стоимость $c_{ij} = C - r_{ij}$, где C обозначает наибольший рейтинг.

Задача о назначениях в терминах теории графов

- Задача о назначениях также может быть описана на языке теории графов, где узлы представляют людей и должности, а ребра представляют рейтинги.

$$R = \begin{bmatrix} 1 & \color{red}{6} & 0 \\ 0 & 8 & \color{red}{6} \\ \color{red}{4} & 0 & 1 \end{bmatrix}$$



- Таким образом, задача Assignment по существу пытается найти Max Weighted Matching в двудольном графе.

Дано: n рабочих, n станков, c_{ij} — время работы i-рабочего на j-м станке.
Найти назначение рабочих на станки с минимальным суммарным временем.

ЦЛП формулировка

- Цель — минимизировать стоимость выполнения работ:

$$\min \sum_{i}^{n} \sum_{j}^{n} w_{ij} x_{ij}$$

ЦЛП формулировка

- Цель — минимизировать стоимость выполнения работ:

$$\min \sum_{i}^{n} \sum_{j}^{n} w_{ij} x_{ij}$$

- Ограничение: работа выполняется одним работником:

$$\sum_{i}^{n} x_{ij} = 1 \quad \forall j = 1, \ldots, n$$

- Цель — минимизировать стоимость выполнения работ:

$$\min \sum_{i}^{n} \sum_{j}^{n} w_{ij} x_{ij}$$

- Ограничение: работа выполняется одним работником:

$$\sum_{i}^{n} x_{ij} = 1 \quad \forall j = 1, \ldots, n$$

- Работник выполняет одну работу:

$$\sum_{j}^{n} x_{ij} = 1 \quad \forall i = 1, \ldots, n$$

ЦЛП формулировка

- Цель — минимизировать стоимость выполнения работ:

$$\min \sum_{i}^{n} \sum_{j}^{n} w_{ij} x_{ij}$$

- Ограничение: работа выполняется одним работником:

$$\sum_{i}^{n} x_{ij} = 1 \quad \forall j = 1, \ldots, n$$

- Работник выполняет одну работу:

$$\sum_{j}^{n} x_{ij} = 1 \quad \forall i = 1, \ldots, n$$

- Все переменные бинарные:
  $x_{ij} \in \{0, 1\}$.

ЦЛП формулировка

Математическая модель

$$\min \sum_{i}^{n} \sum_{j}^{n} w_{ij} x_{ij}$$

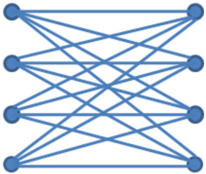
$$\sum_{i}^{n} x_{ij} = 1 \quad \forall j = 1, \ldots, n$$

$$\sum_{j}^{n} x_{ij} = 1 \quad \forall i = 1, \ldots, n$$

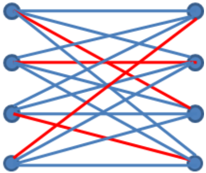
$$x_{ij} \in \{0, 1\}.$$

Пример задачи о назначениях

$$\begin{pmatrix} 2 & 10 & 9 & 7 \\ 15 & 4 & 14 & 8 \\ 13 & 14 & 16 & 11 \\ 4 & 15 & 13 & 19 \end{pmatrix}$$



$$\begin{pmatrix} 2 & 10 & \boxed{9} & 7 \\ 15 & \boxed{4} & 14 & 8 \\ 13 & 14 & 16 & \boxed{11} \\ \boxed{4} & 15 & 13 & 19 \end{pmatrix}$$



Лемма 1

Если веса всех ребер полного двудольного графа, инцидентных какой-либо вершине, изменить (увеличить или уменьшить) на одно и то же число, то для новой задачи совершенное паросочетание наименьшего веса будет состоять из тех же ребер, что и в старой.

Вспомогательные леммы

Лемма 1

Если веса всех ребер полного двудольного графа, инцидентных какой-либо вершине, изменить (увеличить или уменьшить) на одно и то же число, то для новой задачи совершенное паросочетание наименьшего веса будет состоять из тех же ребер, что и в старой.

Доказательство

Совершенное паросочетание для каждой вершины содержит ровно одно ребро, инцидентное этой вершине.
Указанная операция изменит на одно и то же число вес любого паросочетания.
Значит, ребро, которое принадлежало оптимальному паросочетанию в старом графе, в новом графе тоже будет ему принадлежать.

Вспомогательные леммы

Далее будем рассматривать только графы с неотрицательной весовой функцией, так как, согласно лемме 1, задачу о назначениях на остальных графах можно свести к задаче о назначениях на них.

Далее будем рассматривать только графы с неотрицательной весовой функцией, так как, согласно лемме 1, задачу о назначениях на остальных графах можно свести к задаче о назначениях на них.

Лемма 2

Выделим в множествах X и Y подмножества X', Y'. Пусть $d = \min\{c(xy) | x \in X \setminus X', y \in Y'\}$.

Далее будем рассматривать только графы с неотрицательной весовой функцией, так как, согласно лемме 1, задачу о назначениях на остальных графах можно свести к задаче о назначениях на них.

Лемма 2

Выделим в множествах X и Y подмножества X', Y'. Пусть $d = \min\{c(xy) | x \in X \setminus X', y \in Y'\}$.
Прибавим d ко всем весам ребер, инцидентных вершинам из X'.

Вспомогательные леммы

Далее будем рассматривать только графы с неотрицательной весовой функцией, так как, согласно лемме 1, задачу о назначениях на остальных графах можно свести к задаче о назначениях на них.

Лемма 2

Выделим в множествах X и Y подмножества X', Y'. Пусть $d = \min\{c(xy) | x \in X \setminus X', y \in Y'\}$.
Прибавим d ко всем весам ребер, инцидентных вершинам из X'.
Затем отнимем d от всех весов ребер, инцидентных вершинам из Y' (далее для краткости эта операция обозначается как $X' \uparrow\downarrow Y'$).

Вспомогательные леммы

Далее будем рассматривать только графы с неотрицательной весовой функцией, так как, согласно лемме 1, задачу о назначениях на остальных графах можно свести к задаче о назначениях на них.

Лемма 2

Выделим в множествах X и Y подмножества X', Y'. Пусть $d = \min\{c(xy) | x \in X \setminus X', y \in Y'\}$.
Прибавим d ко всем весам ребер, инцидентных вершинам из X'.
Затем отнимем d от всех весов ребер, инцидентных вершинам из Y' (далее для краткости эта операция обозначается как $X' \uparrow\downarrow Y'$).
Тогда:

Далее будем рассматривать только графы с неотрицательной весовой функцией, так как, согласно лемме 1, задачу о назначениях на остальных графах можно свести к задаче о назначениях на них.

Лемма 2

Выделим в множествах X и Y подмножества X', Y'. Пусть $d = \min\{c(xy)|x \in X \setminus X', y \in Y'\}$.
Прибавим d ко всем весам ребер, инцидентных вершинам из X'.
Затем отнимем d от всех весов ребер, инцидентных вершинам из Y' (далее для краткости эта операция обозначается как $X' \uparrow\downarrow Y'$).
Тогда:
Веса всех ребер графа останутся неотрицательными.

Вспомогательные леммы

Далее будем рассматривать только графы с неотрицательной весовой функцией, так как, согласно лемме 1, задачу о назначениях на остальных графах можно свести к задаче о назначениях на них.

Лемма 2

Выделим в множествах X и Y подмножества X', Y'. Пусть $d = \min\{c(xy) | x \in X \setminus X', y \in Y'\}$.
Прибавим d ко всем весам ребер, инцидентных вершинам из X'.
Затем отнимем d от всех весов ребер, инцидентных вершинам из Y' (далее для краткости эта операция обозначается как $X' \uparrow\downarrow Y'$).
Тогда:
Веса всех ребер графа останутся неотрицательными.
Веса ребер вида xy, где $x \in X'$, $y \in Y'$ или $x \in X \setminus X'$, $y \in Y \setminus Y'$, не изменятся.

Вспомогательные леммы

Доказательство

Рассмотрим матрицу весов графа. Не умаляя общности, можно сказать, что множества X' и Y' состоят из первых элементов множеств X и Y соответственно (мы упорядочиваем множества по номерам вершин).

Доказательство

Рассмотрим матрицу весов графа. Не умаляя общности, можно сказать, что множества X' и Y' состоят из первых элементов множеств X и Y соответственно (мы упорядочиваем множества по номерам вершин).
Тогда вся матрица делится на 4 блока:

	Y′	Y\Y′
X′	A+d–d	C+d
X\X′	B–d	D

Доказательство

Рассмотрим матрицу весов графа. Не умаляя общности, можно сказать, что множества X' и Y' состоят из первых элементов множеств X и Y соответственно (мы упорядочиваем множества по номерам вершин).
Тогда вся матрица делится на 4 блока:

	Y′	Y\Y′
X′	A+d−d	C+d
X\X′	B−d	D

Веса группы A будут сначала увеличены, а потом уменьшены на d, поэтому они не изменятся, веса группы D вообще изменяться не будут.

Вспомогательные леммы

Доказательство

Рассмотрим матрицу весов графа. Не умаляя общности, можно сказать, что множества X' и Y' состоят из первых элементов множеств X и Y соответственно (мы упорядочиваем множества по номерам вершин).
Тогда вся матрица делится на 4 блока:

	Y′	Y\Y′
X′	A+d−d	C+d
X\X′	B−d	D

Веса группы A будут сначала увеличены, а потом уменьшены на d, поэтому они не изменятся, веса группы D вообще изменяться не будут.
Все веса группы B будут уменьшены на d, но d — минимум среди этих весов, поэтому они останутся неотрицательными.

Вспомогательные леммы

Лемма 3

Если веса всех ребер графа неотрицательны и некоторое совершенное паросочетание состоит из ребер нулевого веса, то оно является оптимальным.

Лемма 3

Если веса всех ребер графа неотрицательны и некоторое совершенное паросочетание состоит из ребер нулевого веса, то оно является оптимальным.

Доказательство

Действительно, паросочетание с какими-то другими весами ребер имеет больший вес и оптимальным не является.

Венгерский алгоритм

1. Вычитаем из каждой строки значение ее минимального элемента. Теперь в каждой строке есть хотя бы один нулевой элемент.

Венгерский алгоритм

1. Вычитаем из каждой строки значение ее минимального элемента.
   Теперь в каждой строке есть хотя бы один нулевой элемент.
2. Вычитаем из каждого столбца значение его минимального элемента.
   Теперь в каждом столбце есть хотя бы один нулевой элемент.

Венгерский алгоритм

1. Вычитаем из каждой строки значение ее минимального элемента.
   Теперь в каждой строке есть хотя бы один нулевой элемент.
2. Вычитаем из каждого столбца значение его минимального элемента.
   Теперь в каждом столбце есть хотя бы один нулевой элемент.
3. Ищем в текущем графе совершенное паросочетание из ребер нулевого веса:

Венгерский алгоритм

1. Вычитаем из каждой строки значение ее минимального элемента.
   Теперь в каждой строке есть хотя бы один нулевой элемент.
2. Вычитаем из каждого столбца значение его минимального элемента.
   Теперь в каждом столбце есть хотя бы один нулевой элемент.
3. Ищем в текущем графе совершенное паросочетание из ребер нулевого веса:

   if оно найдено, то желаемый результат достигнут, **return**.

Венгерский алгоритм

1. Вычитаем из каждой строки значение ее минимального элемента.
   Теперь в каждой строке есть хотя бы один нулевой элемент.
2. Вычитаем из каждого столбца значение его минимального элемента.
   Теперь в каждом столбце есть хотя бы один нулевой элемент.
3. Ищем в текущем графе совершенное паросочетание из ребер нулевого веса:

   if оно найдено, то желаемый результат достигнут, **return**.

   else покроем нули матрицы весов минимальным количеством строк и столбцов (это не что иное, как нахождение минимального вершинного покрытия в двудольном графе).

Венгерский алгоритм

1. Вычитаем из каждой строки значение ее минимального элемента.
   Теперь в каждой строке есть хотя бы один нулевой элемент.
2. Вычитаем из каждого столбца значение его минимального элемента.
   Теперь в каждом столбце есть хотя бы один нулевой элемент.
3. Ищем в текущем графе совершенное паросочетание из ребер нулевого веса:

   if оно найдено, то желаемый результат достигнут, **return**.

   else покроем нули матрицы весов минимальным количеством строк и столбцов (это не что иное, как нахождение минимального вершинного покрытия в двудольном графе).

   Пусть X_c и Y_c — множества вершин минимального вершинного покрытия из левой и правой долей (то есть, строк и столбцов) соответственно, тогда применим преобразование $X_c \uparrow\downarrow (Y \setminus Y_c)$.

1. Вычитаем из каждой строки значение ее минимального элемента.
   Теперь в каждой строке есть хотя бы один нулевой элемент.
2. Вычитаем из каждого столбца значение его минимального элемента.
   Теперь в каждом столбце есть хотя бы один нулевой элемент.
3. Ищем в текущем графе совершенное паросочетание из ребер нулевого веса:

   if оно найдено, то желаемый результат достигнут, **return**.

   else покроем нули матрицы весов минимальным количеством строк и столбцов (это не что иное, как нахождение минимального вершинного покрытия в двудольном графе).

   Пусть X_c и Y_c — множества вершин минимального вершинного покрытия из левой и правой долей (то есть, строк и столбцов) соответственно, тогда применим преобразование $X_c \uparrow\downarrow (Y \setminus Y_c)$.

   Для этого преобразования d будет минимумом по всем ребрам между $X \setminus X_c$ и $Y \setminus Y_c$, то есть, ребер нулевого веса здесь нет, поэтому, после его выполнения в матрице весов появится новый нуль.

   Перейти к шагу 1.

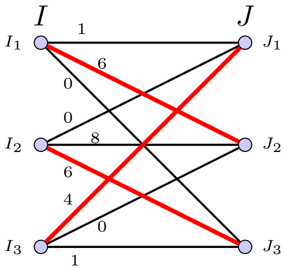
Анализ времени работы

- Поиск максимального паросочетания или минимального вершинного покрытия в двудольном графе совершается за $O(n^3)$ операций.
- При каждом повторении шагов 1-4 в матрице весов появляется новый нуль.
- Этот нуль соответствует некоторому новому ребру между вершинами из множеств $X \setminus X_c$ и $Y \setminus Y_c$.
- Всего в графе n^2 ребер, значит, всего будет совершено не более $O(n^2)$ итераций внешнего цикла.
- Поэтому, верхняя оценка времени работы данного метода — $O(n^5)$.
- Временная сложность оригинального алгоритма была $O(n^4)$, однако Эдмондс и Карп показали, что его можно модифицировать так, чтобы достичь времени выполнения $O(n^3)$.
- Форд и Фалкерсон распространили метод на общие транспортные задачи.

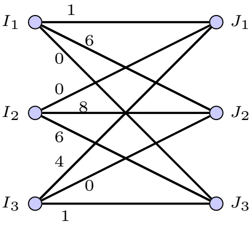
Задача о назначениях на максимум

ВХОД: полный двудольный граф $G = (I \cup J, E)$, каждому ребру (i, j) приписан вес r_{ij}.
ВЫХОД: **совершенное паросочетание** M, имеющее наибольший суммарный вес.

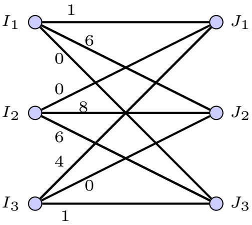
$$R = \begin{bmatrix} 1 & 6 & 0 \\ 0 & 8 & 6 \\ 4 & 0 & 1 \end{bmatrix}$$



ЦЛП формулировка задачи MaxWeightedMatching

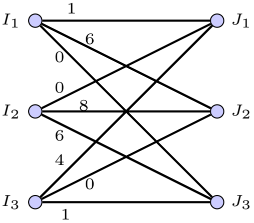


ЦЛП формулировка задачи MaxWeightedMatching



- Из-за весов ребер метод сетевого потока не работает, как в задаче MaxMatching. Теперь вернемся к ЦЛП формулировке.

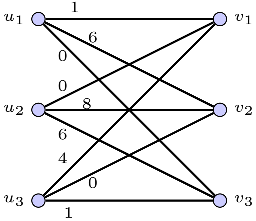
ЦЛП формулировка задачи MaxWeightedMatching



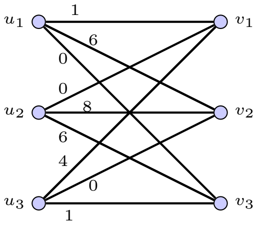
- Из-за весов ребер метод сетевого потока не работает, как в задаче MaxMatching. Теперь вернемся к ЦЛП формулировке.
- Прямая задача:

$$
\begin{array}{llrcll}
\max & \sum_{i\in I}\sum_{j\in J} & r_{ij}x_{ij} & & & \\
s.t. & \sum_{i\in I} & x_{ij} & = & 1 & \text{для всех } j\in J \\
 & \sum_{j\in J} & x_{ij} & = & 1 & \text{для всех } i\in I \\
 & & x_{ij} & = & 0/1 & \text{для всех } i\in I, j\in J
\end{array}
$$

Двойственная задача: MinWeightedVertexCover

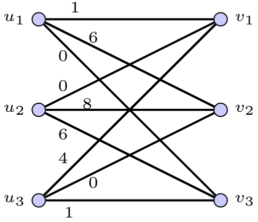


Двойственная задача: MinWeightedVertexCover



- Матрица коэффициентов прямой задачи полностью унимодулярна; таким образом, мы можем заменить $x_{ij} = 0/1$ на $1 \geq x_{ij} \geq 0$ и написать двойственную задачу (задачу о минимальном взвешенном покрытии вершин).

Двойственная задача: MinWeightedVertexCover



- Матрица коэффициентов прямой задачи полностью унимодулярна; таким образом, мы можем заменить $x_{ij} = 0/1$ на $1 \geq x_{ij} \geq 0$ и написать двойственную задачу (задачу о минимальном взвешенном покрытии вершин).
- Двойственная задача:

$$\begin{array}{lrcrcll} \min & \sum_{i \in I} u_i & + & \sum_{j \in J} v_j & & & \\ s.t. & u_i & + & v_j & \geq & r_{ij} & \text{для всех } (i,j) \end{array}$$

Критерий оптимальности для X и (U, V)

- Прямая задача:

$$\begin{array}{llrlll}
\max & \sum_{i\in I}\sum_{j\in J} & r_{ij}x_{ij} & & & \\
s.t. & \sum_{i\in I} & x_{ij} & = & 1 & \text{для всех } j \in J \\
 & \sum_{j\in J} & x_{ij} & = & 1 & \text{для всех } i \in I \\
 & & x_{ij} & \leq & 1 & \text{для всех } i \in I, j \in J
\end{array}$$

Критерий оптимальности для X и (U, V)

- Прямая задача:

$$\begin{array}{llrcll}
\max & \sum_{i\in I}\sum_{j\in J} & r_{ij}x_{ij} & & & \\
s.t. & \sum_{i\in I} & x_{ij} & = & 1 & \text{для всех } j\in J \\
 & \sum_{j\in J} & x_{ij} & = & 1 & \text{для всех } i\in I \\
 & & x_{ij} & \le & 1 & \text{для всех } i\in I, j\in J
\end{array}$$

- Двойственная задача:

$$\begin{array}{lrcrcll}
\min & \sum_{i\in I} u_i & + & \sum_{j\in J} v_j & & & \\
s.t. & u_i & + & v_j & \ge & r_{ij} & \text{для всех ребер } (i,j)
\end{array}$$

Критерий оптимальности для X и (U,V)

- Прямая задача:

$$\begin{array}{llcll}
\max & \sum_{i\in I}\sum_{j\in J} r_{ij}x_{ij} & & & \\
s.t. & \sum_{i\in I} x_{ij} & = & 1 & \text{для всех } j\in J \\
 & \sum_{j\in J} x_{ij} & = & 1 & \text{для всех } i\in I \\
 & x_{ij} & \le & 1 & \text{для всех } i\in I, j\in J
\end{array}$$

- Двойственная задача:

$$\begin{array}{llcl}
\min & \sum_{i\in I} u_i + \sum_{j\in J} v_j & & \\
s.t. & u_i + v_j & \ge & r_{ij} \quad \text{для всех ребер } (i,j)
\end{array}$$

- ортогональность: утверждает, что прямая и двойственная задачи имеют одинаковые значения целевых функций.

$$(u_i + v_j - r_{ij})x_{ij} = 0 \text{ для всех } (i,j)$$

Критерий оптимальности для X и (U, V)

- Прямая задача:

$$
\begin{array}{llrcll}
\max & \sum_{i \in I} \sum_{j \in J} & r_{ij} x_{ij} & & & \\
s.t. & \sum_{i \in I} & x_{ij} & = & 1 & \text{для всех } j \in J \\
 & \sum_{j \in J} & x_{ij} & = & 1 & \text{для всех } i \in I \\
 & & x_{ij} & \leq & 1 & \text{для всех } i \in I, j \in J
\end{array}
$$

- Двойственная задача:

$$
\begin{array}{llclcll}
\min & \sum_{i \in I} u_i & + & \sum_{j \in J} v_j & & & \\
s.t. & u_i & + & v_j & \geq & r_{ij} & \text{для всех ребер } (i, j)
\end{array}
$$

- ортогональность: утверждает, что прямая и двойственная задачи имеют одинаковые значения целевых функций.

$$(u_i + v_j - r_{ij}) x_{ij} = 0 \text{ для всех } (i, j)$$

- Критерий оптимальности:

Две стратегии решения

- Критерий оптимальности:
  - (1) X — допустимое решение прямой задачи: X представляет совершенное паросочетание.

$$\begin{array}{rcll} \sum_{i\in I} x_{ij} & = & 1 & \text{для всех } j \in J \\ \sum_{j\in J} x_{ij} & = & 1 & \text{для всех } i \in I \\ x_{ij} & \le & 1 & \text{для всех } i \in I, j \in J \end{array}$$

  - (2) (U,V) — допустимое решение двойственной задачи:

$$u_i \quad + \quad v_j \quad \ge \quad r_{ij} \qquad \text{для всех ребер } (i,j)$$

  - (3) X и (U,V) являются ортогональными:

$$(u_i + v_j - r_{ij})x_{ij} = 0 \text{ для всех } (i,j)$$

Две стратегии решения

- Критерий оптимальности:
  - (1) X — допустимое решение прямой задачи: X представляет совершенное паросочетание.

$$\begin{array}{rcll} \sum_{i\in I} x_{ij} & = & 1 & \text{для всех } j \in J \\ \sum_{j\in J} x_{ij} & = & 1 & \text{для всех } i \in I \\ x_{ij} & \le & 1 & \text{для всех } i \in I, j \in J \end{array}$$

  - (2) (U, V) — допустимое решение двойственной задачи:

$$u_i + v_j \ge r_{ij} \quad \text{для всех ребер } (i, j)$$

  - (3) X и (U, V) являются ортогональными:

$$(u_i + v_j - r_{ij})x_{ij} = 0 \text{ для всех } (i, j)$$

- Прямой метод: инициализировать X и (U, V), чтобы они удовлетворяли условиям (1) и (2), и улучшить их, чтобы условие (3) выполнялось.

Две стратегии решения

- Критерий оптимальности:
  - (1) X — допустимое решение прямой задачи: X представляет совершенное паросочетание.

$$\begin{array}{rcll} \sum_{i \in I} x_{ij} & = & 1 & \text{для всех } j \in J \\ \sum_{j \in J} x_{ij} & = & 1 & \text{для всех } i \in I \\ x_{ij} & \leq & 1 & \text{для всех } i \in I, j \in J \end{array}$$

  - (2) (U, V) — допустимое решение двойственной задачи:

$$u_i + v_j \geq r_{ij} \quad \text{для всех ребер } (i, j)$$

  - (3) X и (U, V) являются ортогональными:

$$(u_i + v_j - r_{ij})x_{ij} = 0 \text{ для всех } (i, j)$$

- Прямой метод: инициализировать X и (U, V), чтобы они удовлетворяли условиям (1) и (2), и улучшить их, чтобы условие (3) выполнялось.
- Двойственный метод: инициализировать X и (U, V), чтобы они удовлетворяли условиям (2) и (3), и улучшить их, чтобы условие (1) выполнялось. Это то, что делает венгерский

2.1 Венгерский метод: решения двойственной задачи

Венгерский метод: решения двойственной задачи

- Двойственная задача:

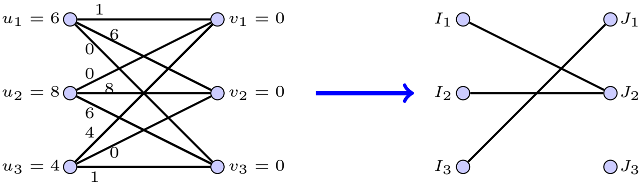
$$\begin{array}{lccccccl} \min & \sum_{i\in I} u_i & + & \sum_{j\in J} v_j & & & & \\ s.t. & u_i & + & v_j & \geq & r_{ij} & & \text{для всех ребер } (i,j) \end{array}$$

Венгерский метод: решения двойственной задачи

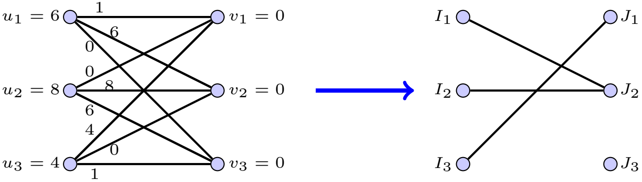
- Двойственная задача:

$$
\begin{array}{lccccccl}
\min & \sum_{i\in I} u_i & + & \sum_{j\in J} v_j & & & \\
s.t. & u_i & + & v_j & \geq & r_{ij} & \text{для всех ребер } (i,j)
\end{array}
$$

- Основная идея: изначально устанавливается (U, V) так чтобы эта пара была двойственным допустимым решением и выполнялась ортогональность X и (U, V), т.е. $(U_i + v_j - r_{ij})x_{ij} = 0$ для всех (i, j). Затем пытаемся сделать X допустимым решением прямой задачи, т.е. в результате X образует совершенное паросочетание.



- Найти двойственное допустимое (U, V) и ортогональное X — тривиально (установить $x_{ij} = 0$, если $u_i + v_j > r_{ij}$). Эгервари и Кун удалили такие ребра и сосредоточились на оставшемся графе (называемом **графом равенств** $G_E(U, V)$).



Положить $u_i = \max_j r_{ij}$;
Положить $v_i = 0$;
while TRUE **do**
    Построить граф равенств $G_E(U,V)$ с единственным ребром (i,j) если $u_i + v_j = r_{ij}$;
    if $G_E(U,V)$ имеет совершенное паросочетание **M then**
        return M;
    end if
    Уменьшить u_i или v_j;
end while

Задача 1. Дан двудольный граф, требуется найти в нём максимальное паросочетание минимального веса (т.е. в первую очередь максимизируется размер паросочетания, во вторую — минимизируется его стоимость).

Задача 1. Дан двудольный граф, требуется найти в нём максимальное паросочетание минимального веса (т.е. в первую очередь максимизируется размер паросочетания, во вторую — минимизируется его стоимость).

- Для решения просто строим задачу о назначениях, ставя на месте отсутствующих рёбер число «бесконечность».

Задача 1. Дан двудольный граф, требуется найти в нём максимальное паросочетание минимального веса (т.е. в первую очередь максимизируется размер паросочетания, во вторую — минимизируется его стоимость).

- Для решения просто строим задачу о назначениях, ставя на месте отсутствующих рёбер число «бесконечность».
- После этого решаем задачу венгерским алгоритмом, и удаляем из ответа рёбра бесконечного веса (они могли войти в ответ, если у задачи нет решения в виде совершенного паросочетания).

Задача 2. Дан двудольный граф, требуется найти в нём паросочетание максимального веса.

Задача 2. Дан двудольный граф, требуется найти в нём паросочетание максимального веса.

- Решение опять же очевидно, только все веса надо умножить на минус единицу (либо в венгерском алгоритме заменить все минимумы на максимумы, а бесконечности — на минус бесконечности).

Задача 3. Детектирование движущихся объектов по снимкам: было произведено два снимка, по итогам которых было получено два набор координат. Требуется соотнести объекты на первом и втором снимке, т.е. определить для каждой точки второго снимка, какой точке первого снимка она соответствовала. При этом требуется минимизировать сумму расстояний между сопоставленными точками (т.е. мы ищем решение, в котором объекты суммарно прошли наименьший путь).

Задача 3. Детектирование движущихся объектов по снимкам: было произведено два снимка, по итогам которых было получено два набор координат. Требуется соотнести объекты на первом и втором снимке, т.е. определить для каждой точки второго снимка, какой точке первого снимка она соответствовала. При этом требуется минимизировать сумму расстояний между сопоставленными точками (т.е. мы ищем решение, в котором объекты суммарно прошли наименьший путь).

- Для решения мы просто строим и решаем задачу о назначениях, где в качестве весов рёбер выступают евклидовы расстояния между точками.

Задача 4. Детектирование движущихся объектов по локаторам:

Задача 4. Детектирование движущихся объектов по локаторам:

- есть два локатора, которые умеют определять не положение объекта в пространстве, а лишь направление на него.

Задача 4. Детектирование движущихся объектов по локаторам:

- есть два локатора, которые умеют определять не положение объекта в пространстве, а лишь направление на него.
- С обоих локаторов (расположенных в различных точках) поступила информация в виде таких направлений.

Задача 4. Детектирование движущихся объектов по локаторам:

- есть два локатора, которые умеют определять не положение объекта в пространстве, а лишь направление на него.
- С обоих локаторов (расположенных в различных точках) поступила информация в виде таких направлений.
- Требуется определить положение объектов, т.е. определить предполагаемые положения объектов и соответствующие им пары направлений так, чтобы минимизировать сумму расстояний от объектов до лучей-направлений.

Задача 4. Детектирование движущихся объектов по локаторам:

- есть два локатора, которые умеют определять не положение объекта в пространстве, а лишь направление на него.
- С обоих локаторов (расположенных в различных точках) поступила информация в виде таких направлений.
- Требуется определить положение объектов, т.е. определить предполагаемые положения объектов и соответствующие им пары направлений так, чтобы минимизировать сумму расстояний от объектов до лучей-направлений.
- Решение — строим и решаем задачу о назначениях, где вершинами первой доли являются n направлений с первого локатора, вершинами второй доли — n направлений со второго локатора, а весами рёбер — расстояния между соответствующими лучами.