ЗАДАЧА О ПОИСКЕ УСТОЙЧИВЫХ ПАРОСОЧЕТАНИЙ

Задача устойчивых паросочетаний была частично сформулирована в **1962** году, когда два экономиста-математика,
Дэвид Гейл и **Ллойд Шепли**, задались вопросом:

Можно ли спланировать процесс поступления в колледж (или приема на работу), который был бы саморегулируемым (self-enforcing)?

Решение было опубликовано в 1962 году в журнале American Mathematical Monthly в статье под названием «Поступление в колледж и стабильность браков».

Элвин Рот разработал очень много практических механизмов, основанных на алгоритме Гейла-Шелли.

Ллойд Шепли и Элвин Рот в 2012 году получили **Нобелевскую премию** по экономике «За теорию стабильного распределения и практики устройства рынков».

Примеры механизмов, которые были внедрены:

- Распределение врачей по больницам
- Распределение интернов по больницам
- Набор спортсменов в команды
- Набор стажеров в компании
- Наем клерков в суды
- Подбор школ для детей
- Доноры и реципиенты

Группа студентов колледжа начинает подавать заявки в компании на летнюю практику.

В процессе обработки заявок важно взаимодействие двух разных сторон: компаний (нанимателей) и студентов (кандидатов).

- Каждый кандидат упорядочивает список компаний в порядке своих предпочтений.
- Каждая компания - после поступления заявок - формирует свой порядок предпочтений для кандидатов, подавших заявки.

На основании этих предпочтений компании обращаются с предложениями к некоторым из своих кандидатов.

Кандидаты решают, какое из полученных предложений стоит принять.

Возможные сбои

Радж только что принял предложение от крупной телекоммуникационной компании CluNet. Через несколько дней начинающая компания WebExodus, которая тянула с принятием нескольких окончательных решений, связывается с Раджем и тоже предлагает ему летнюю практику. Вообще-то, с точки зрения Раджа, вариант с WebExodus предпочтительнее CluNet - скажем, из-за непринужденной атмосферы и творческого азарта. Этот поворот заставляет Раджа отказаться от предложения CluNet и пойти в WebExodus. Лишившись практиканта, CluNet предлагает работу одному из запасных кандидатов, который мгновенно отменяет свое предыдущее согласие на предложение мегакорпорации Babelsoft.

Возможные сбои

Подруге Раджа по имени Челси было назначено отправиться в Babelsoft. Но услышав историю Раджа, она звонит в WebExodus и говорит: «Знаете, я бы предпочла провести это лето в вашей фирме, а не в Babelsoft». Отдел кадров WebExodus охотно верит; более того, заглянув в заявку Челси, они понимают, что она перспективнее другого студента, у которого уже запланирована летняя практика. И если компания WebExodus не отличается особой деловой принципиальностью, она найдет способ отозвать свое предложение другому студенту и возьмет Челси на его место.

Основная проблема

Процесс **не является саморегулируемым**.

Если участникам разрешено произвольно действовать, исходя из их собственных интересов, весь процесс может быть нарушен.

Многие участники - как кандидаты, так и наниматели - могут оказаться недовольны как самим процессом, так и его результатом.

Формулировка задачи с устойчивыми результатами

Можно ли для имеющегося набора предпочтений по кандидатам и нанимателям распределить кандидатов по нанимателям так, чтобы для каждого нанимателя E и каждого кандидата A, который не был принят на работу к E, выполнялось **по крайней мере одно** из следующих двух условий?

1. Каждый из кандидатов, принятых E на работу, с его точки зрения, предпочтительнее A.
2. С точки зрения A, его текущая ситуация предпочтительнее работы на нанимателя E.

Другие источники происхождения задачи

За десять лет до работы Гейла и Шепли очень похожая задача использовалась Национальной программой распределения студентов-медиков по больницам.

Более того, эта система с относительно незначительными изменениями продолжает применяться и в наши дни.

Упрощенная постановка задачи

Каждый из n кандидатов подает заявки в каждую из n компаний, а каждая компания хочет принять на работу одного кандидата.

Частный случай задачи

Имеется множество $M = \{m_1, ..., m_n\}$ из n мужчин и множество $W = \{w_1, ..., w_n\}$ из n женщин.

Паросочетание (марьяж) S представляет собой множество пар из $M \times W$, обладающее тем свойством, что каждый элемент M и каждый элемент W встречается <u>не более чем в одной</u> паре в S.

Идеальным паросочетанием S' называется паросочетание, при котором каждый элемент M и каждый элемент W встречается <u>ровно в одной</u> паре из S'

Понятие предпочтений

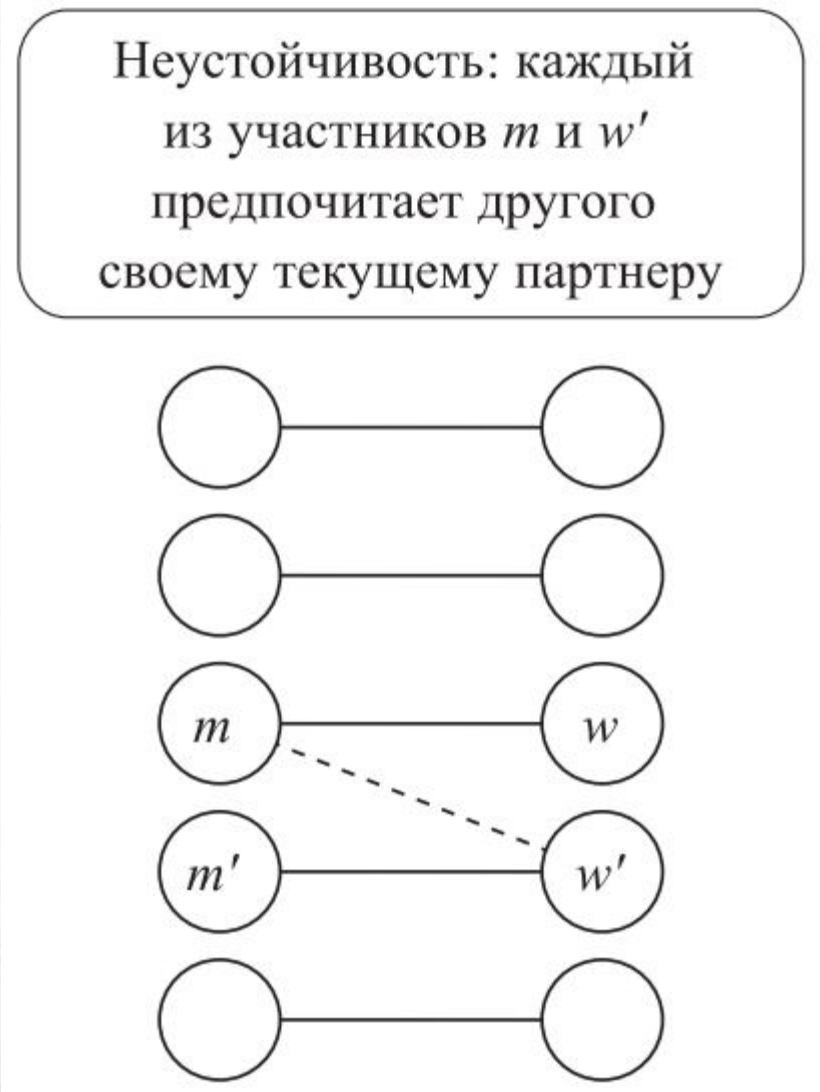
Каждый мужчина $m \in M$ формирует оценки всех женщин; мы говорим, что m предпочитает w женщине w', если m присваивает w более высокую оценку, чем w'.

Мы будем называть упорядоченную систему оценок m его списком предпочтений.

«Ничьи» в оценках запрещены.

Аналогичным образом каждая женщина назначает оценки всем мужчинам.

Проблемы, имеющиеся в идеальном паросочетании



Цель: создать паросочетание без неустойчивых пар.

Паросочетание S называется **устойчивым**, если оно (1) идеально и (2) не содержит неустойчивости в отношении S.

Вопросы:

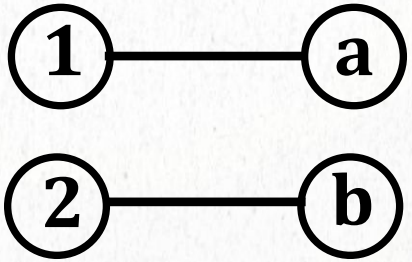
- Существует ли устойчивое паросочетание для каждого набора списков предпочтений?
- Можно ли эффективно построить устойчивое паросочетание для имеющегося списка предпочтений (если оно существует)?

Пример 1

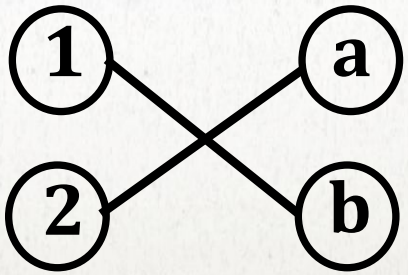
Мужчины: {1, 2}
Женщины: {a, b}
Предпочтения:

$$\begin{array}{c} 1 \\ 2 \end{array}\begin{pmatrix} a & b \\ a & b \end{pmatrix} \qquad \begin{array}{c} a \\ b \end{array}\begin{pmatrix} 1 & 2 \\ 1 & 2 \end{pmatrix}$$

Идеальное, но неустойчивое паросочетание



устойчивое паросочетание



Пример 2

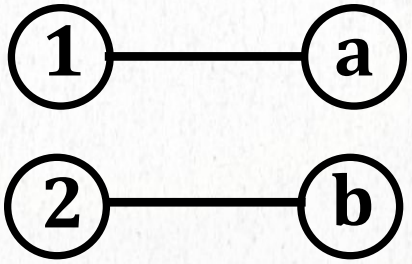
Мужчины: {1, 2}

Женщины: {a, b}

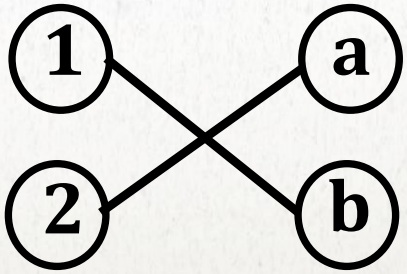
Предпочтения:

$$\begin{array}{c} 1 \\ 2 \end{array}\begin{pmatrix} a & b \\ b & a \end{pmatrix} \qquad \begin{array}{c} a \\ b \end{array}\begin{pmatrix} 2 & 1 \\ 1 & 2 \end{pmatrix}$$

1-ое устойчивое паросочетание



2-ое устойчивое паросочетание



Пример 3

Мужчины: {1, 2, 3, 4, 5}

Женщины: {a, b, c, d, e}

Предпочтения:

Мужчины:

1 :	b	c	a	e	d
2 :	a	b	c	e	d
3 :	b	e	a	d	c
4 :	e	c	d	a	b
5 :	a	c	e	d	b

Женщины:

a :	2	3	4	5	1
b :	3	4	5	1	2
c :	5	1	3	2	4
d :	2	4	1	5	2
e :	3	1	2	5	4

Мужчины делают предложения, а женщины выбирают.

<u>Шаг 1.</u> Начальные предложения. Мужчины делают предложения (первый столбец матрицы предпочтений), женщины в соответствии с приоритетом выбирают.

1 → b
2 → a
3 → b
4 → e
5 → a

a	b	c	d	e
2 ~~5~~	~~1~~ 3			4

<u>Шаг 2.</u> Отвергнутые мужчины делают новые предложения (второй столбец матрицы предпочтений).

1 → c

5 → c

a	b	c	d	e
2	3	~~1~~ 5		4

<u>Шаг 3.</u> Настойчивый мужчина 1 делает очередное предложение женщине **a** и отвергнут ею.

1 → a

a	b	c	d	e
2 ~~1~~	3	5		4

<u>Шаг 4.</u> И вновь мужчина 1. Его предложение женщиной **e** принято, а мужчина 4 ею отвергнут.

1 → e

a	b	c	d	e
2	3	5		~~4~~ 1

<u>Шаг 5.</u> Отвергнутый мужчина 4 делает новое предложение.

4 → c

a	b	c	d	e
2	3	5 ~~4~~		1

<u>Шаг 6.</u> Новое предложение мужчины 4.

4 → d

a	b	c	d	e
2	3	5	4	1

В итоге получаем **устойчивое паросочетание**:

$$1 \to e,\ 2 \to a,\ 3 \to b,\ 4 \to d,\ 5 \to c$$

Предложения делают женщины, а мужчины осуществляют выбор.

<u>Шаг 1.</u> Начальные предложения женщин.

a → 2
b → 3
c → 5
d → 2
e → 3

1	2	3	4	5
	a ~~d~~	b ~~e~~		c

<u>Шаг 2.</u> Отвергнутые женщины делают новые предложения.

d → 4

e → 1

1	2	3	4	5
e	a	b	d	c

Получили **устойчивое паросочетание**, причем оно совпадает с тем, когда предложения делали мужчины, а женщины выбирали.

Глобальные структуры данных

man, women : **Array** [1..n, 1..n] **Of Integer;** {Матрицы предпочтений}

indexMan : **Array** [1.. n] **Of Integer;** {Номер предложения i-го мужчины}

married : **Array** [1.. n] **Of Integer;** {Результат, married[i] определяет номер мужчины, с которым женщина i сочетается законным браком}

freeman : **Array** [1..n] **Of Boolean;** {Признак занятости мужчин. Если freeman[i]=**True**, то мужчина с номером i свободен}

Начальная инициализация данных:

```
For i := 1 To n Do Begin
 married[i] := -1; {Женщины не заняты}
 indexMan[i] := 1; {Каждый мужчина
делает предложение первой женщине из
своего списка предпочтений}
 freeMan[i] := True; {Мужчины свободны}
End;
```

Основная логика:

```
Procedure Solve;
 var i, cw : Integer;{i - № мужчины, cw - № женщины}
 Begin
  While Not Result Do {Пока не найдено паросочетание}
   For i := 1 To n Do
    If freeMan[i] Then Begin {i-ый мужчина свободен}
      cw := man[i, indexMan[i]];
      WriteLn('мужчина ', i, ' делает предложение ', cw, '
женщине');
      If (married[cw] = -1) Then Begin {Женщина свободна}
          married[cw] := i;
          freeMan[i] := False;
         End
       Else SelectW(i,cw); {Женщина занята и вынуждена
делать выбор}
    End;
 End;
```

Проверка того, что найдено устойчивое паросочетание на этих структурах данных осуществляется подсчетом количества сформировавшихся пар. Если оно равно значению n, то паросочетание найдено.

```
Function Result : Boolean;
 Var i, k : Integer;
 Begin
  k := 0;
  For i := 1 To n Do
     If (married[i] <> -1) Then k := k+1;
  If k = n Then Result := True
   Else Result := False;
 End;
```

Логика процедуры выбора женщины:

Идем по приоритетам женщины cw и ищем, кто встретится раньше: ее жених на данный момент или сделавший только что предложение i-ый мужчина.

```
Procedure SelectW(i, cw: Integer);
{i - № мужчины, cw - № женщины}
 Var j : Integer;{№ приоритета у женщины cw}
     pp : Boolean;{женщина cw сделала выбор}
 Begin
  j := 1;
  pp := False;

 End;
```

Цикл по j

```
While (j <= n) And Not pp Do Begin
    If (women[cw, j] = married[cw]) Then Begin {жених в приоритете}
        indexMan[i] := indexMan[i] + 1; {i-му мужчине надо выбирать следующую женщину по его приоритету}
        pp := True;
      end
     Else If (women[cw, j] = i) Then Begin {Женщина отдает предпочтение мужчине с номером i}
      indexMan[married[cw]] := indexMan[married[cw]] + 1; {жениху женщины cw надо выбирать следующую женщину по его приоритету}
      freeMan[married[cw]] := True; {жених  становится свободным}
      freeMan[i] := False; {i-ый мужчина становится занятым}
      married[cw] := i;{i становится женихом для cw}
      pp := True;
     End;
    j :=j + 1;
   End;
```

Задача 1

Мужчины: {1, 2, 3}

Женщины: {a, b, c}

Предпочтения:

Мужчины:

1 :	a	b	c
2 :	a	b	c
3 :	a	b	c

Женщины:

a :	3	2	1
b :	1	3	2
c :	2	1	3

Задача 2

Мужчины: {1, 2, 3, 4,}

Женщины: {a, b, c, d}

Предпочтения:

Мужчины:

1 :	a	b	c	d
2 :	a	b	c	d
3 :	a	b	c	d
4 :	a	b	c	d

Женщины:

a :	3	4	2	1
b :	3	1	4	2
c :	2	4	1	3
d :	1	3	2	4