Министерство образования и науки Украины

Севастопольский национальный технический университет

МЕТОДИЧЕСКИЕ УКАЗАНИЯ К ПРАКТИЧЕСКИМ ЗАНИЯТИЯМ

по дисциплине «Элементы теории конечных автоматов и графов»

Часть 2

для студентов специальности 07080401
«Компьютеризированные системы обработки информации и управления»

ст. преп. кафедры Информационных систем
к.т.н., Ю.В. Доронина, асс. Деркунская В.О.

Севастополь, 2010

УДК 004.92

Элементы теории конечных автоматов. Методические указания для выполнения практических и контрольных работ по дисциплине “Теория конечных автоматов и графов” для студентов специальности 7.080401 "Информационные управляющие системы и технологии" /Сост. Ю.В. Доронина. Деркунская В.О. - Севастополь: Изд-во СевНТУ, 2010.-22 с.

Цель методических указаний: выработка у учащихся практических навыков по проектированию и анализу технических устройств на базе модели конечных автоматов.

Методические указания рассмотрены и утверждены на заседании кафедры Информационных Систем.
Протокол № 1 от 17 февраля 2010 г.

Рецензент: доцент кафедры ПЭОТ канд.техн.наук., Г.В. Тараненко

Занятие 5. Часть 5.2.
Минимизация таблицы переходов и выходов(ТПиВ)
(Минимизация числа состояний конечного автомата)

Дискретные автоматы **эквивалентны**, если, начиная с состояния S_0, при любой последовательности они работают одинаково.
Задача минимизации – задача уменьшения числа состояний автомата.

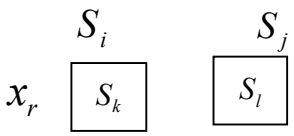
Для полностью определенного автомата эта задача решается однозначно (т.е. существует только один автомат с минимальным числом состояний).

Si		**Sj**	**Sv**
Si	…	Sj	Sv
Sj	…	Si	Sn
x	…	x	Sf
Sl	…	Si	Sw
Yi		**Yj**	**Yv**

Si → Sv – несовместимы, если Sl и Sw неэквивалентны. При условии, что Yi=Yv. Состояния Si → Sj совместимы (если Yi=Yj).
Эти свойства приводят к тому, что при минимизации мы можем получить различное число тупиковых решений. Обычно задачи минимизации решаются методом перебора.

Если состояния $S_i u \quad S_j$ не являются неэквивалентными, то для проверки их эквивалентности необходимо проверить эквивалентность следующих состояний $S_k u \quad S_l$, в которые автомат переходит под воздействием одного и того же сигнала x_r. Тогда говорят, что состояния $S_i u \quad S_j$ влекут на проверку эквивалентности состояний $S_k u \quad S_l$.

$S_{ij} \rightarrow S_{kl}$:



Состояния $S_i u \quad S_j$ могут повлечь несколько влекомых пар, соответствующих разным входным наборам.

Влекомые пары могут снова создавать влекомые пары. Таким образом, получается дерево влекомых пар.
Если получается влекомая пара, которая уже есть, то ее не записывают.
Если какая-нибудь влекомая пара неэквивалентна, то можно объявить неэквивалентными все влекомые пары по соответствующей цепочке влекомых пар.
Если новых влекомых пар не появилось, нет неэквивалентных пар, то все пары в этом дереве эквивалентны, кроме того, эквивалентны пары, которые могут быть получены с помощью свойства транзитивности: $S_{ji} \rightarrow S_{jk} \Rightarrow S_{jk}$

Алгоритм минимизации ТПиВ

1) Строится матрица отношения эквивалентностей. Столбцы сопоставляются $S_1,....,S_l$, строки - $S_0,....,S_{l-1}$ и рисуется первая половина этой матрицы.
2) В матрицу записываются 0 (нули) в те элементы, которые тождественно неэквивалентны и 1(единицы) – в элементы, которые эквивалентны. Остальные остаются незаполненными.
3) Выбирается очередной незаполненный элемент S_0S_2 (или S_2S_0) и строится дерево влекомых пар. Если появляется невлекомая пара, то вся цепочка – неэквивалентна.
4) Если таблица полностью заполнена – переход к п.5, иначе п.3.
5) Из матрицы выписываются совокупности эквивалентных состояний следующим образом: берется очередная не вычеркнутая строка матрицы, и выписываются состояния, которые есть в этой строке и где есть единицы. Далее смотрят, вошло ли S_l в какую либо группу. Если нет, то формируют новую отдельную совокупность, и процесс закончен.

ПРИМЕР 1

S0	**S1**	**S2**	**S3**
S0	S0	S0	S3
S1	S1	S1	S3
S2	S2	S2	S0
S2	S2	S3	S0
1	**0**	**1**	**1**

S1	S2	S3	
0			S0
	0	0	S1
			S2

Определяют априорно (изначально несовместимые состояния. Анализ априорно несовместимых состояний производится по выходным сигналам.) Чтобы отличить априорно несовместимые состояния, они либо подчеркиваются либо отмечаются другим знаком.

ПРИМЕР 2

							SS	
S0	S1	S2	S3	S4	S5	S6	7	SS8
S0	S1	S6	S5	S4	S5	S6	S7	S8
S1	S1	S8	S8	S7	S0	S0	S7	S0
S2	S3	S4	S4	S2	S1	S7	S3	S7
S0	S2	S1	S7	S3	S2	S3	S3	S3
0	0	1	1	0	1	1	0	1

S1	S2	S3	S4	S5	S6	S7	S8	
0	0	0	0	0	0	0	0	S0
	0	0	1	0	0	1	0	S1
		1	0	0	0	0	0	S2
			0	0	0	0	0	S3
				0	0	1	0	S4
					1	0	1	S5
						0	1	S6
							0	S7

Нэ ← Нэ

$S_0S_1 \rightarrow S_0S_2, S_0S_4 \rightarrow S_0S_4$

Нэ ← Нэ

$S_0S_7 \to S_0S_3$, $S_1S_4 \to S_1S_7$,

Нэ ← Нэ

нэ $S_2S_3 \to S_5S_6$,

$S_2S_5 \to S_5S_6$ **нэ**

↘ S_0S_8 , $S_2S_6 \to S_0S_8$, $S_3S_5 \to S_0S_8$, $S_3S_6 \to S_0S_8$

нэ **нэ**

$S_3S_8 \to S_5S_8$

↘ S_0S_8

и т.п. Аналогично проверяются на совместимость все состояния (то есть заполняются все строки треугольной матрицы).

Класс эквивалентности $\{S_0\}, \{S_1S_4S_7\}, \{S_2S_3\}, \{S_5S_6S_8\}$

4 состояния S_0 S_I S_{II} S_{III}

Для построения новой минимизированной таблицы необходимо сопоставить каждому множеству эквивалентных состояний новое состояние. Состояние S_I - берется любое состояние, например, S_I.

Алгоритм построения минимизированной таблицы

1) Полученные при преобразовании классы эквивалентности обозначаются новыми состояниями (для удобства римскими цифрами – не забывать иначе обозначать начальное состояние!).

2) Строится таблица, число строк которой равно числу строк исходной таблицы, а число столбцов – количеству полученных состояний. Если есть еще незаполненный столбец, то берется одно из состояний, относящихся к классу эквивалентности данного нового состояния и заполняется столбец по старой таблице переходов и выходов, осуществляя замену старых состояний на новые, к классу которого относится старое состояние.

3) Выходные функции сохраняются. Снова п.2. Если все столбцы заполнены, то процедура заканчивается.

	$S_\Diamond$	S_I	S_{II}	S_{III}
$S_\Diamond$	$S_\Diamond$	S_I	S_{III}	S_{III}
S_I	S_I	S_I	S_{III}	S_{III}
S_{II}	S_{II}	S_{II}	S_I	S_I
$S_\Diamond$	$S_\Diamond$	S_{II}	S_I	S_I
	0	0	1	1

РЕШЕНИЕ ЗАДАЧ.

1) Минимизировать ТПиВ

	S_0	S_1	S_2	S_3	S_4	S_5	S_6	S_7	S_8
x_1	0	1	1	1	0	1	1	0	1
x_2	S_0	S_1	S_6	S_5	S_4	S_5	S_6	S_7	S_8
	S_1	S_1	S_8	S_8	S_7	S_0	S_0	S_7	S_0
	S_0	S_2	S_1	S_7	S_3	S_2	S_3	S_3	S_3
	S_2	S_3	S_4	S_4	S_2	S_1	S_7	S_3	S_7

2)Минимизировать ТПиВ

	S_0	S_1	S_2	S_3	S_4	S_5	S_6	S_7	S_8
x_1	0	0	1	0	0	1	1	0	1
x_2	S_0	S_1	S_6	S_5	S_4	S_5	S_6	S_7	S_8
	S_1	S_1	S_8	S_8	S_7	S_0	S_0	S_7	S_0
	S_0	S_2	S_1	S_7	S_3	S_2	S_3	S_3	S_3
	S_2	S_3	S_4	S_4	S_2	S_1	S_7	S_3	S_7

3)Минимизировать ТПиВ

	S_0	S_1	S_2	S_3	S_4	S_5	S_6	S_7	S_8
x_1	0	0	1	0	1	1	1	0	0
x_2	S_0	S_1	S_6	S_5	S_4	S_5	S_6	S_7	S_8
	S_1	S_1	S_8	S_8	S_7	S_0	S_0	S_7	S_0
	S_0	S_2	S_1	S_7	S_3	S_2	S_3	S_3	S_3
	S_2	S_3	S_4	S_4	S_2	S_1	S_7	S_3	S_7

4) Минимизировать ТпиВ

x_2 \ x_1	1	1	0	0	1	0	1	0	0
	S_0	S_1	S_2	S_3	S_4	S_5	S_6	S_7	S_8
	S_0	S_1	S_6	S_5	S_4	S_5	S_6	S_7	S_8
	S_1	S_1	S_8	S_8	S_7	S_0	S_0	S_7	S_0
	S_0	S_2	S_1	S_7	S_3	S_2	S_3	S_3	S_3
	S_2	S_3	S_4	S_4	S_2	S_1	S_7	S_3	S_7

Домашнее задание: Повторить минимизацию ТП и В. Подготовиться к контрольной работе по теме.

Занятие 6.
Часть 6.1.
Примитивная таблица П и В асинхронного автомата

Примитивная таблица – таблица в столбцах которой есть одно только одно устойчивое состояние.

Используется при исходном построении таблицы П и В АА. Переход от словесного к математическому описанию алгоритма его работы (чаще всего она имеет лишние состояния, громоздка), но не содержит никаких излишних сложностей.

Рассмотрим пример построения ПТП и В для автомата типа «счетчик».

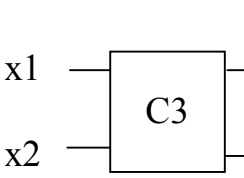
Сразу 2 сигнала одновременно изменяться не может.

Моделирование:

	x_1	x_2	Q_2	Q_1
S_0	0	0	S_0	0	0
	0	1	S_2	0	1
	1	0	S_1	1	1
S_1	0	1	S_1	1	1
	0	0	S_4	1	1
	1	1	S_3	0	0

S_4	S_5	S_6	S_7	S_8	S_9	S_{10}	S_{11}	S_{12}
S_4	S_5	S_{10}	S_0	S_0	S_{10}	S_{10}	S_5	S_4
S_6	S_8	S_6	x	S_8	x	S_{11}	S_{11}	x
x	x	S_3	S_3	S_3	S_3	x	S_3	S_3
S_7	S_9	x	S_7	x	S_9	S_{12}	x	S_{12}

Пример2 Модель «Секретный замок»:



y1 = { 1, замок открыт; 0, замок закрыт }

y2 = { 1, тревога звучит; 0, тревога не звучит }

открывает послед.

x_1	0 1 1
x_2	1 1 0

сброс тревоги

x_1	1 0 1
x_1	0 0 0

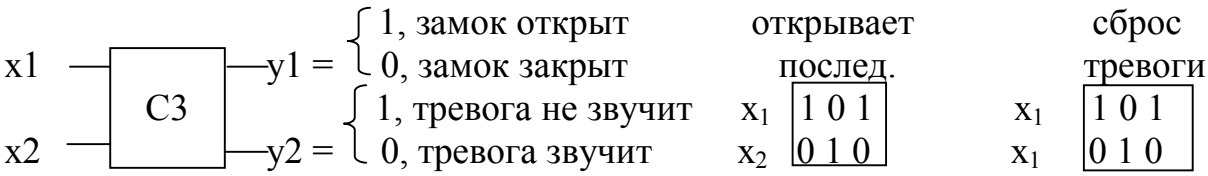
S_0	S_1	S_2	S_3	S_4	S_5	S_6	S_7	S_8	S_9	S_{10}	S_{11}	S_{12}
S_0	S_3	S_3	S_3	x	x	S_9^1	S_3	S_{10}	S_9	S_{10}	S_0	S_0	S_0
S_1	S_1	x	S_6	S_6	S_8	S_6^2	x	S_8	S_{11}	S_{12}	S_{11}	S_{12}	x
x	S_4	S_5	x	S_4	S_5		S_4	x	x	x	x	x	x
S_2	x	S_2	S_7	S_7	S_7		S_7	x	S_7	S_{13}	x	x	S_{13}
00	01	00	01	01	00	01	01	x0	01	10	0x	x0	x0

-1

+1

РЕШЕНИЕ ЗАДАЧ:

Построить ПТПиВ для конечного автомата: «Секретный замок»:



Домашнее задание: Проанализировать ошибки самостоятельной работы, изучить алгоритм построения ПТПиВ.

Часть 6. 2
Самостоятельная работа по теме:
«Построение ТП и В по размеченному регулярному выражению. Минимизация ТПиВ»

Занятие 7.
Часть 7.1
Анализ результатов Самостоятельной работы. Работа над ошибками.
Часть 7.2
ГРАММАТИКИ И КОНЕЧНЫЕ АВТОМАТЫ

Зачем нужны автоматы?

Многие объекты связаны с интуитивным понятием автоматического выполнения действий. Принято называть **автоматом** объект, который выполняет некоторые, как правило «осмысленные» действия самостоятельно, без внешнего вмешательства. Например, автомат, сортирующий некоторые виды сырья, распределяющий далее эти виды по отдельным контейнерам, применяющий для каждого сырья специальную технологию обработки и осуществляющий упаковку готовой продукции. Такой автомат получает исходное сырье, чтобы *повторить цикл* своей работы согласно *внутренней логике* (программе).

Естественной и понятной явилась попытка построить общие *математические модели автоматического поведения*. Для оценки их значения достаточно понять, что *компьютер – это универсальный автомат*, перерабатывающий исходную информацию по *изменяемой программе*. Математические идеи, возникшие на пути создания моделей автоматов, вскоре привели к достаточно сложным абстрактным понятиям и построению содержательной теории.

Стоит заметить, что с интуитивным понятием автомата связаны некоторые *последовательности действий* и *дискретные пошаговые процессы*. «Зная», какое действие за каким следует выполнять, автомат управляет сам собой по внутренней программе.

В центре теории автоматов – задачи синтеза (построения) автоматов, удовлетворяющих заданным условиям.

В книге Халл Э., Джексон К., Дик Д. «Разработка и управление требованиями. Практическое руководство пользователя»
(http://download.telelogic.com/download/article/eBook_RU_Requirements_Engineering.pdf)

Приведена диаграмма состояний для полета самолета, которая размещена ниже.

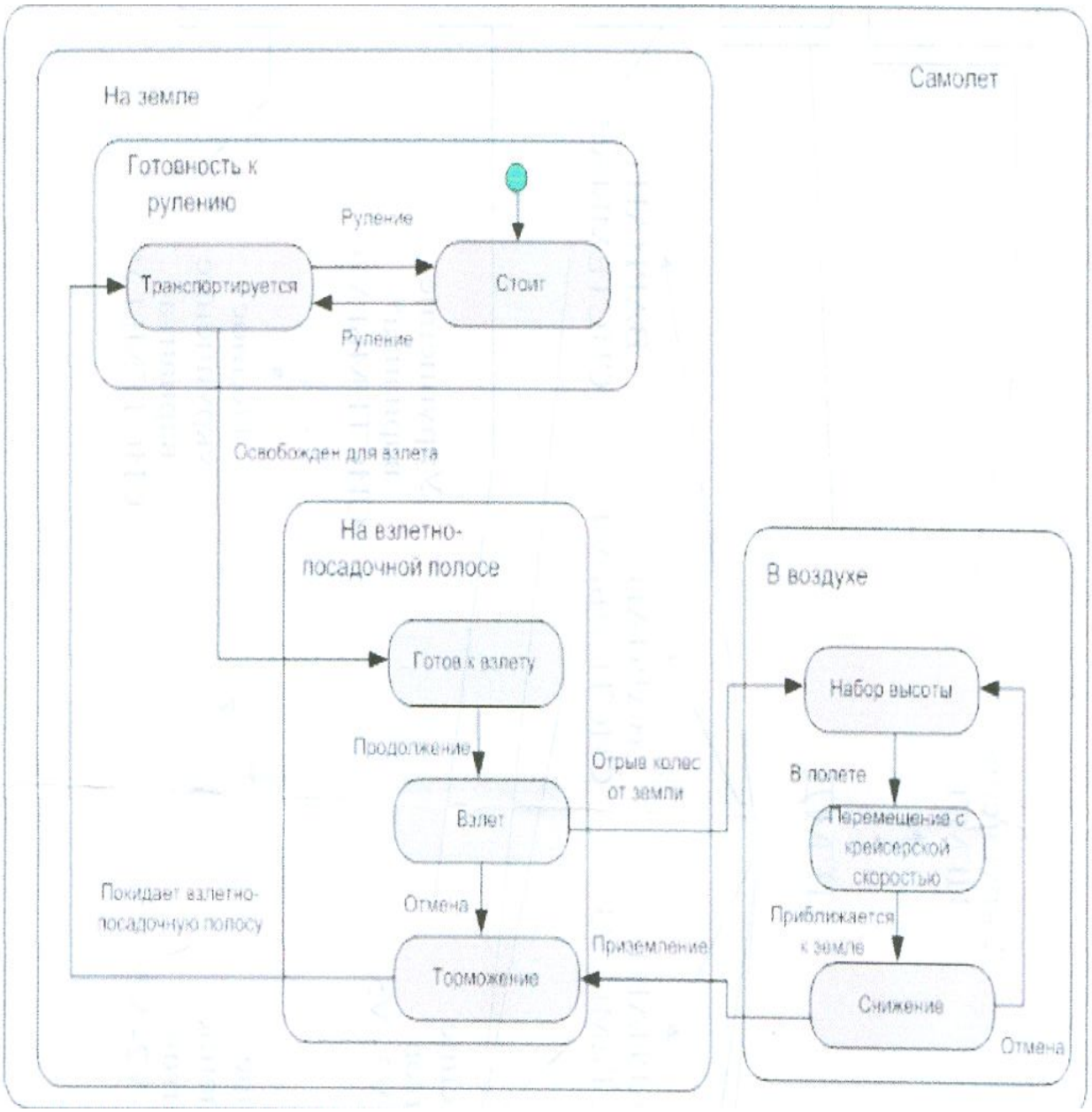


Рисунок 1 – Граф переходов автомата управления самолетом на верхнем уровне.

Вопрос: можно ли описать верхний уровень алгоритма управления самолетом проще и понятнее, чем приведенным здесь графом переходов конечного автомата. Ответ – нельзя. Особенно, если учесть тот факт, что устройств, описываемых аналогично масса (принтеры, кофеварки, турникеты и т.д.), а преобразование графов переходов в программе не представляет никакого труда.

Графы не только тесно взаимосвязаны с конечными автоматами, они являются существенным элементом математических моделей в самых разнообразных областях науки и практики. Они помогают наглядно представить взаимоотношения между объектами или событиями в сложных системах.

Грамматика

Грамматика представляет собой совокупность 4•х объектов:

$$G = (N, T, P, S).$$

N - конечное множество нетерминальных символов языка (словарь нетерминалов);
T - конечное множество терминальных символов языка (словарь терминалов),
P - конечное множество продукций (правил подстановки) вида $\alpha \to \beta$,

где $\alpha \in (N \cup T)^+, \beta \in (N \cup T)^*$,

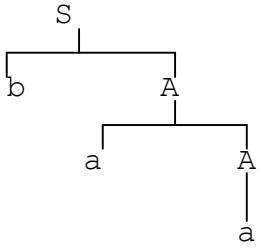
+ - отсутствуют пустые строки, * - есть пустые строки.
S - начальный символ языка (аксиома).

1) Пример правил подстановки,
S -> bA|A
A -> a|aA
где: | - или, - (альтернатива),
S,A – нетерминалы,
a,b - терминалы,
-> - это есть,
S - аксиома.
Цепочки ими порождаемые
суть a, aa, a...a,
ba,baa, baa…a.

Существуют лево- н правосторонние формы вывода, отражающие порядок применения правил грамматики при порождении цепочек.
S=>bA=>baA=>baa - левосторонняя форма вывода, (иногда обозначают S=> *baa).

Предложение - строка, в которой все символы терминальные.
Левосторонняя форма вывода от правосторонней отличается тем, что раскрывается самый левый символ.

Процесс вывода удобно представлять в виде ориентированного графа, который называется *синтаксическим деревом* или *деревом вывода*. Для рассмотренной сентенциальной формы оно имеет вид:



РЕШЕНИЕ ЗАДАЧ:

1) Построить дерево вывода, используя правила подстановки:

```
S -> bA|BA|A
A -> a|aA|aB
```

2) Построить дерево вывода, используя правила подстановки:

```
S -> bA|BAa|aA|BA|a

A -> aA|aAB|a
```

3) Построить дерево вывода, используя правила подстановки:

```
S -> AaAb|bBbAa|bAba
A -> ab|aA|aBb
```

Грамматика является однозначной, если для каждой сентенциальной формы существует единственное синтаксическое дерево, т.е., любая сентенциальная форма выводится следующим образом. Подчеркнем, что порядок применения правил несущественен.

Пример 1.

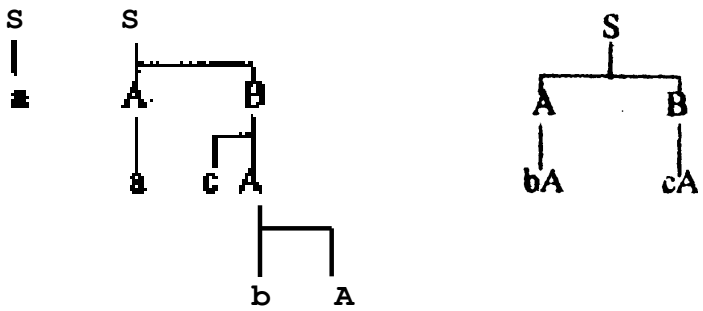
Определить, однозначны ли грамматики?

```
a)  S->AB|a
    A->bA|a
    B->cA
```

Решение.

Для ответа на вопрос, поставленный в задаче, воспользуемся определением.

a)



Грамматика однозначна, порождаемые цепочки суть

{a, aca, acb…ba, bb…bacb…ba}

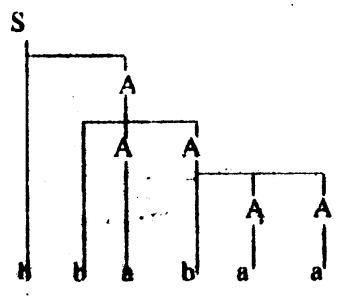
Пример 2.
Определить, однозначны ли грамматики?

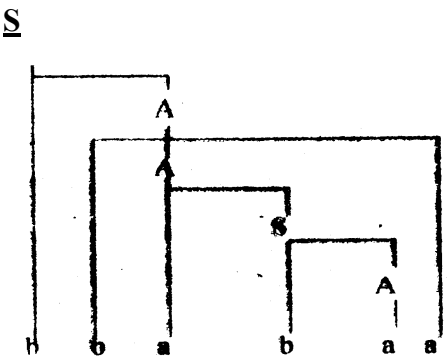
```
a) S-> bA|aB
   A->a|aS|bAA
   B->b|bS|aBB
```

b) S->aB|bAS|bA
   A->bAA|a
   B->ABB|b

Решение.

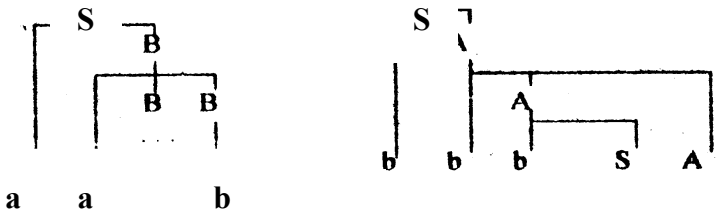
a)





Следовательно,грамматика неоднозначна.

б)



Грамматика однозначна.

РЕШЕНИЕ ЗАДАЧ:

1.Определить, однозначны ли грамматики?

a) S-> bA|S
   A->b|a|bAA
   B->a|bS|aB

b) S->a|bASABa|bAB
   A->bAA|a|abA
   B->ABB|b

2.Определить, однозначны ли грамматики?
a) S-> A|S|SBa|aSbA
   A->a|bAA
   B->a|bS

3.Построить дерево вывода
a) S-> bA|Sa|a
   A->b|a|AaB|ABaB|A
   B->a|bSa|AB

Занятие 8.
Элементы теории графов

Обзор алгоритмов, изученных в курсе ОДМ.

Обыкновенным графом называется пара G=(V,E), где V – конечное множество, E – множество неупорядоченных пар различных элементов из V.

Элементы множества V называют *вершинами* графа, элементы множества E – его *ребрами.*

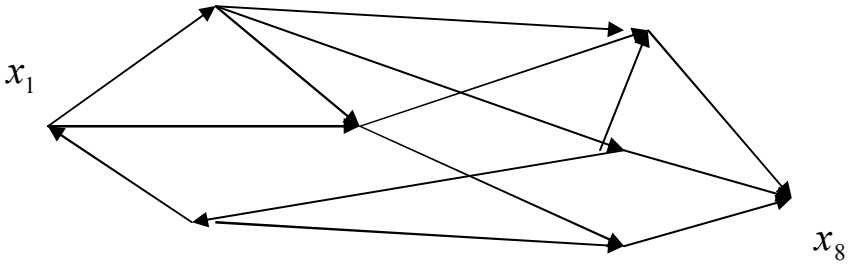
Ориентированным графом называется пара G=(V,E), где V – конечное множество, E – множество упорядоченных пар различных элементов из V. Ориентированный граф часто называют *орграфом***.**

Взвешенный граф – граф обозначенный весами дуг или ребер.

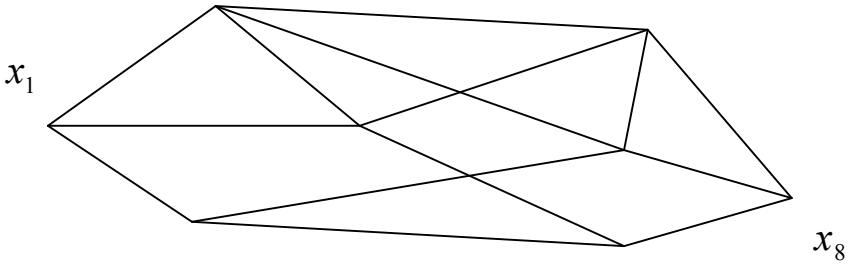
Расстоянием между двумя вершинами графа называется *наименьшая длинна пути*, соединяющая эти вершины. Расстояние между вершинами обозначается через d (a,b). Если в графе нет пути, соединяющего a и b, то есть эти вершины принадлежат разным компонентам связности, то расстояние между ними считается *бесконечным*.

РЕШЕНИЕ ЗАДАЧ:

1. Найти кратчайший путь в графе, используя волновой алгоритм.



2. Найти минимальный путь в графе, установив произвольные веса ребер.



Связность графов

Компоненты связанности

Выделение компонент связанности в ненаправленном графе:

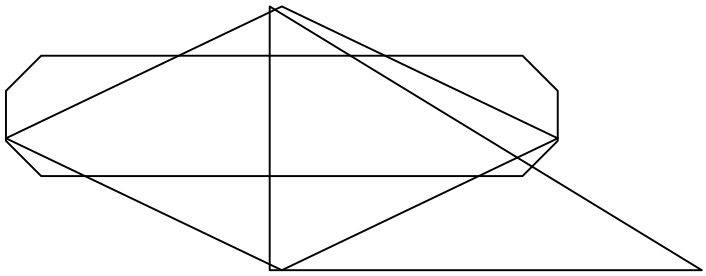
1) Все вершины объявляются не взвешенными, i=0. Выбираем очередную не взвешенную вершину и всем вершинам связанным с ней присваиваем i+1.
2) Отмечаем индексом i вершины ранее не отмеченные. Распространяем i+1 до всех сними связанных.
3) Множество вершин i является подграфом с номером i, если есть вершина, то пункт 2, иначе конец.

Выделение компонент связанности в направленном графе.

1) Сведения направленного графа к некоторому частному не направленному графу (не нарушая свойства сильной связанности и убирая дуги, обеспечиваем слабую связанность).
2) Используется алгоритм, описанный выше.

РЕШЕНИЕ ЗАДАЧ:

1) Выделить компоненты связанности графе:



Циклы в графе. Система независимости циклов.

Контур начинается и заканчивается в одной точке и есть *цикл*.
В графе может быть несколько циклов. Циклу сопоставляется M - мерный двоичный вектор, где M множество вершин графа.
Система циклов графа, каждый из которых нельзя получить линейной комбинацией других циклов, **линейно независимы**.
Между векторами можно вводит операцию сложения. Система независимых циклов, это максимально линейно независимая система циклов.

Теорема Эйлера о числе независимых циклов в графе:

В любом не направленном графе число независимых циклов вычисляется по формуле:

$$v = m - n + p \quad (*)$$

m-число ребер;
n-число вершин;
p-компонента связанности.

Плоские графы

Плоскими* или *планарными называются графы, для которых существует такое изображение на плоскости, у которых ребро не пересекается.

Теоремы:

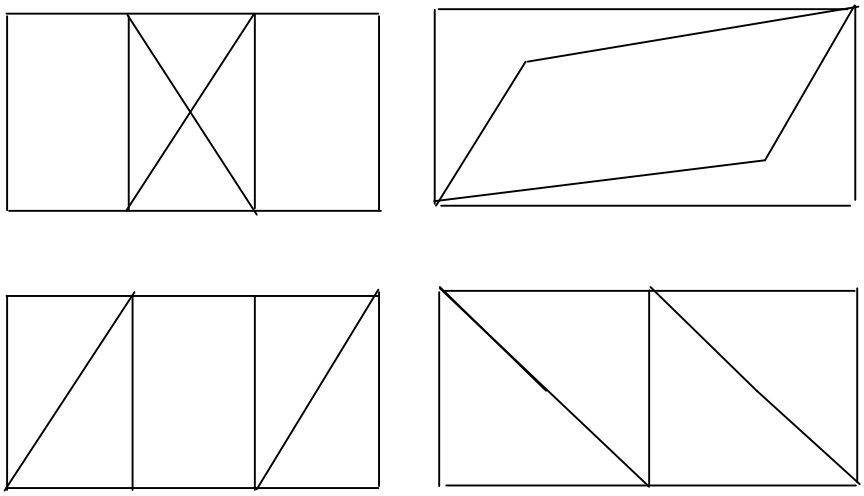
1) Если имеем плоское изображение графа, то оно разбивает плоскость на части (несколько конечных и одна бесконечная). Каждая часть сопоставляется гранью.
2) **Теорема о циклах:**

Края конечных граней плоского графа составляет систему независимых циклов.

3) **Теорема о 4-х красках:**

Плоский граф имеет число $\chi \leq 4$

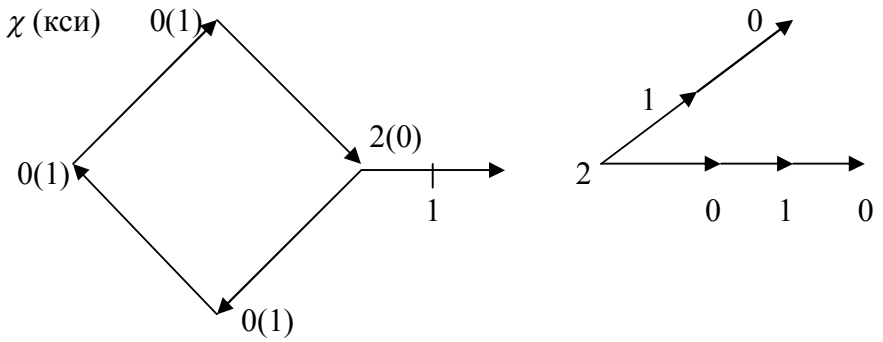
ЗАДАНИЯ:
Для каждого из четырех графов ответить на следующие вопросы:
а) есть ли в нем циклы Эйлера? Если есть, то привести пример, если нет, то сказать, почему;
б) есть ли в нем пути Эйлера? Если есть, то привести пример, если нет, то сказать, почему;
в) есть ли в нем циклы Гамильтона? Если есть, то привести пример;
г) есть ли в нем пути Гамильтона? Если есть, то привести пример;



Задача о раскраске вершин графов

Надо раскрасить вершины графа в различные цвета, так что бы, так что бы смежные вершины не были закрашены одной краской, при этом надо потратить меньше краски.
X- минимальное количество краски.

Функция Гранди.



Теорема:

Если в графе нет цикла нечетной длины, то X для него ≤ 2.

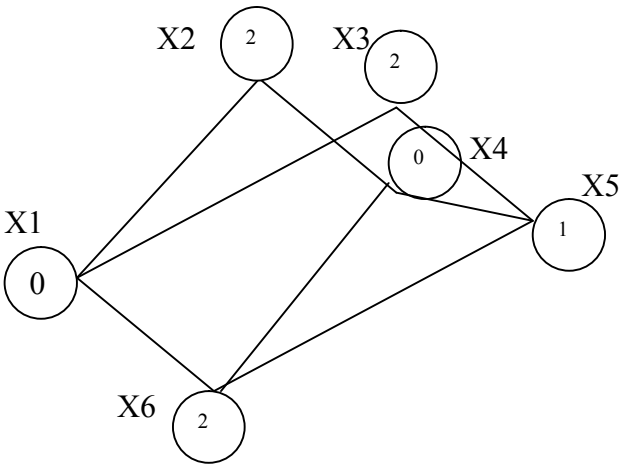
Утверждение:

Вершины внутреннего устойчивого множества можно всегда покрасить одной краской, это максимальное множество вершины, которое можно покрасить одной краской.

Алгоритм кратчайшей раскраски вершин графа

1) Для исходного графа строится множество внутренних устойчивых подмножеств вершин графа.
2) Решается задача о кратчайшем покрытии.
3) Если вершина попадает в несколько выбранных решений, то в одном она остается, из других удаляется. Вершины из каждого устойчивого множества окрашиваются в один цвет, из разных множеств в разные.

	X1	X2	X3	X4	X5	X6	
1)	1			(1)			{X1,X4}0
2)	1				1		{X1,X5}1
3)		1	1			(1)	{X2,X3,X6}2
4)		1			1		
5)			1	1			



Занятие 9.

Зачетное занятие.

Заключительная самостоятельная работа.

Проставление зачетов.