Лекция 03.
Проблема грамматического разбора. Распознаватели.

Разбор цепочек. Задача разбора

- **Задача разбора в общем виде**: на основе имеющейся грамматики некоторого языка **построить распознаватель** для этого языка

- Цепочка принадлежит языку, порождаемому грамматикой, только в том случае, если существует ее **вывод** из аксиомы этой грамматики.
- Процесс построения такого вывода (а, следовательно, и определения принадлежности цепочки языку) называется **разбором**.

Выводы в грамматике

- **Определение:**
вывод цепочки b ∈ (**VT**)* из S ∈ **VN** в КС-грамматике G = (**VT**, **VN**, **P**, S), называется *левым (левосторонним)*, если в этом выводе каждая очередная сентенциальная форма получается из предыдущей заменой самого левого нетерминала.

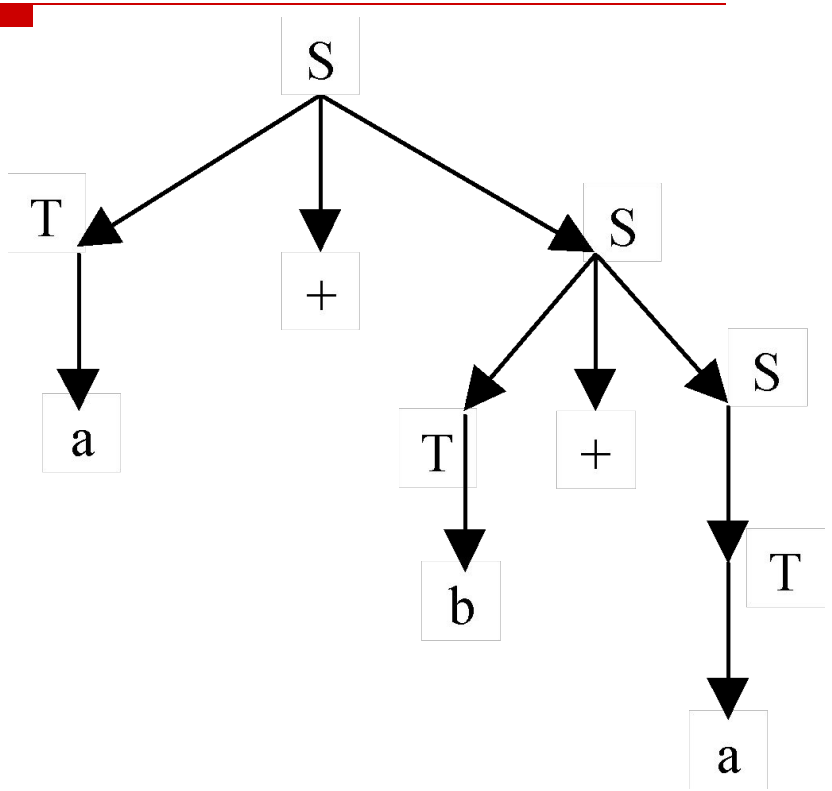
- **Определение:**
вывод цепочки b ∈ (**VT**)* из S ∈ **VN** в КС-грамматике G = (**VT**, **VN**, **P**, S), называется *правым (правосторонним)*, если в этом выводе каждая очередная сентенциальная форма получается из предыдущей заменой самого правого нетерминала.

Пример. Выводы

- Для цепочки a+b+a в грамматике
  G = ({a,b}, {S,T}, {S → T | T+S; T → a|b}, S)
  можно построить выводы:
- Левый
  S→T+S→T+T+S→T+T+T→a+T+T→a+b+T→a+b+a
- Правый
  S→T+S→a+S→a+T+S→a+b+S→a+b+T→a+b+a
- Смешанный - не левый, не правый
  S→T+S→T+T+S→T+T+T→T+T+a→T+b+a→a+b+a

Дерево вывода для цепочки a+b+a

Определение:
КС-грамматика G называется *неоднозначной*, если существует хотя бы одна цепочка a ∈ **L**(G), для которой может быть построено два или более различных деревьев вывода.



В противном случае грамматика называется *однозначной*.

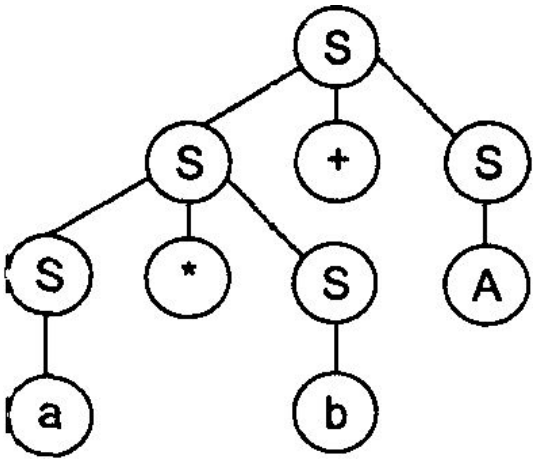
Пример неоднозначной грамматики

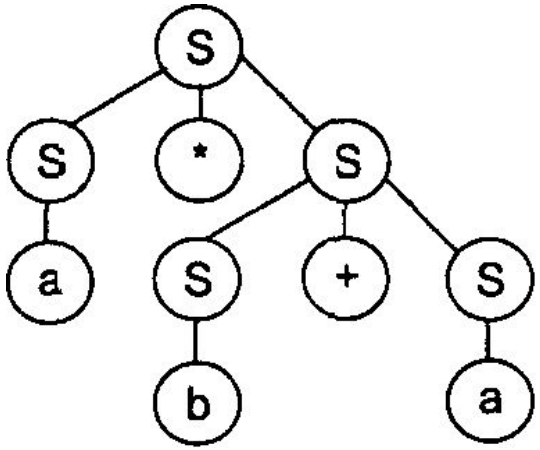
- G = ({if, then, else, a, b}, {S}, **P**, S),
  где **P** = {S → if b then S else S | if b then S | a}.
- В этой грамматике для цепочки
  if b then if b then a else a
  можно построить два дерева вывода.

- Неоднозначность – свойство грамматики, а не языка
- S → if b then S | if b then S’ else S | a
  S’ → if b then S’ else S’ | a

Еще раз вернемся к неоднозначным грамматикам

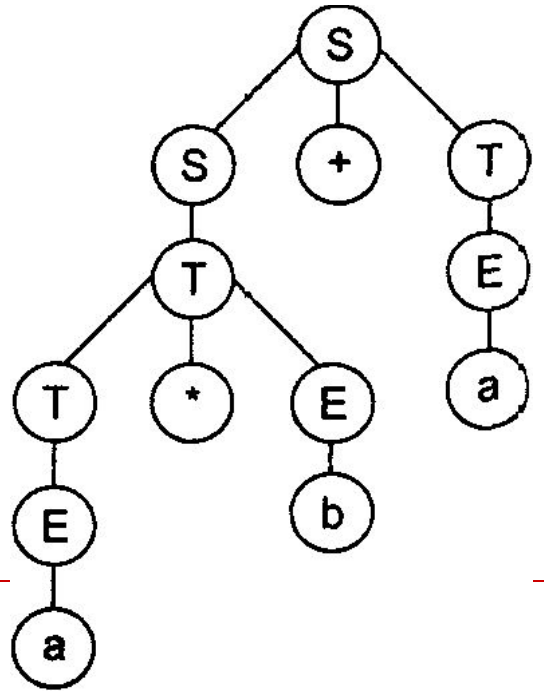
- G({+,-,*,/,(,),a,b},{S},**P**,S):
  P: S → S+S | S-S | S*S | S/S | (S) | a | b
- Для цепочки a*b+a возможны два левых вывода:
  (1) S ⇒ S+S ⇒ S*S+S ⇒ a*S+S ⇒ a*b+S ⇒ a*b+a
  (2) S ⇒ S*S ⇒ a*S ⇒ a*S+S ⇒ a*b+S ⇒ a*b+a





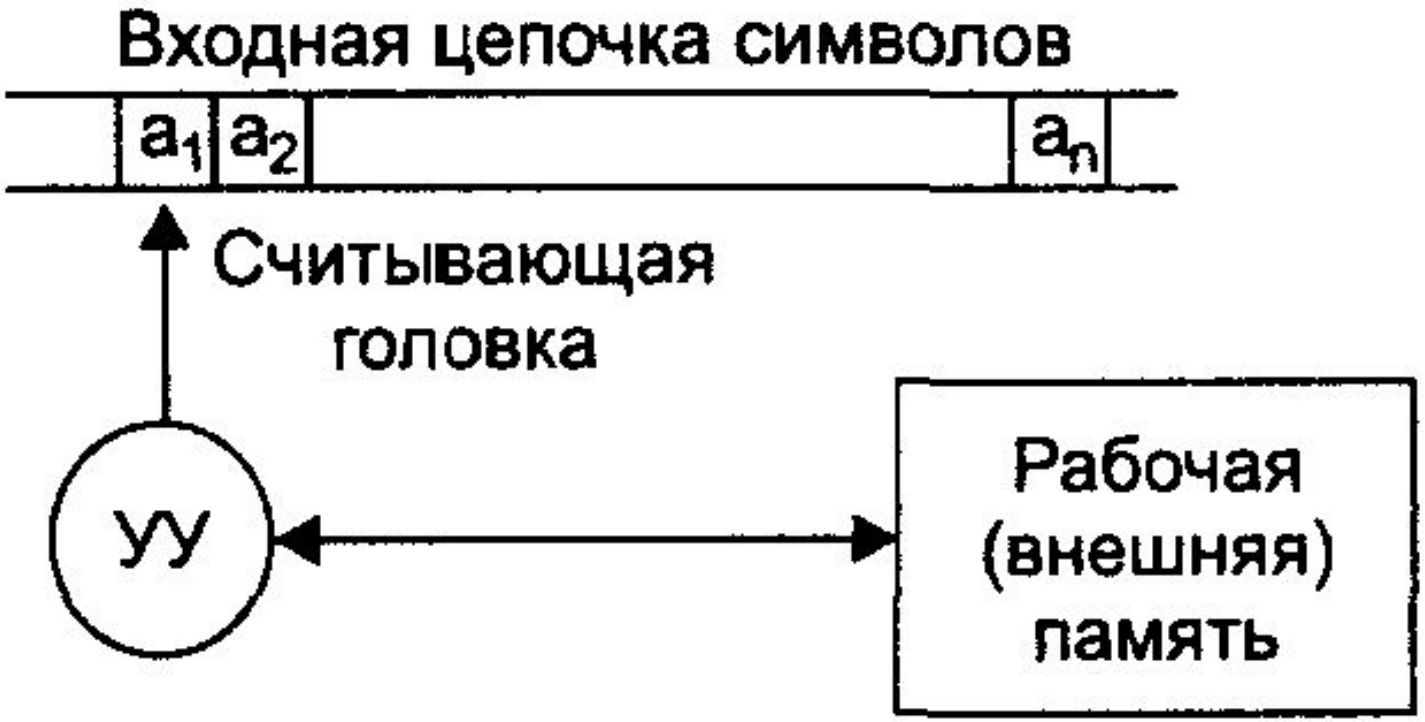
Построение эквивалентной однозначной грамматики

- G'({+,-,*./,(,),a,b},{S,T,E},P',S);
  P‘ = {S → S+T | S-T | T ; T → T*E | T/E | E ;
  E → (S) | a | b}
- Для цепочки a*b+a возможен только один левый вывод:
  S ⇒ S+T ⇒ T+T ⇒ T*E+T ⇒
  E*E+T ⇒ a*E+T ⇒ a*b+T ⇒
  a*b+E ⇒ a*b+a

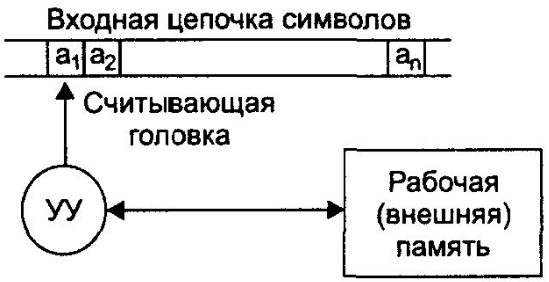


Распознаватели

Распознаватели.
Условная схема распознавателя



Компоненты распознавателя



- **лента**, содержащая исходную цепочку входных символов, и **считывающая головки**, обозревающей очередной символ в этой цепочке;
- **устройство управления** (УУ), которое координирует работу распознавателя, имеет некоторый набор состояний и конечную память (для хранения своего состояния и некоторой промежуточной информации);
- **внешняя (рабочая) память**, которая может хранить некоторую информацию в процессе работы распознавателя и в отличие от памяти УУ может иметь неограниченный объем

- Алфавит распознавателя – ленты + внутренний
- Операции распознавателя – чтение/запись

Работа распознавателя состоит из *последовательности тактов*

- В начале каждого такта состояние распознавателя определяется его **конфигурацией**.
- В процессе работы конфигурация меняется.
- *Конфигурация* определяется:
  - содержимым входной цепочки символов и положением считывающей головки в ней;
  - состоянием УУ;
  - содержимым внешней памяти.

- Всегда задается *начальная конфигурация.*
  И несколько *конечных конфигураций*

Работа распознавателя. Язык, определенный распознавателем

- В начальной конфигурации:
  - считывающая головка на первом символе входной цепочки,
  - УУ находится в заданном начальном состоянии,
  - внешняя память пуста.
- В конечной конфигурации
  - считывающая головка - за концом исходной цепочки
- Распознаватель *допускает входную цепочку* а, если,
  - находясь в начальной конфигурации и
  - получив на вход эту цепочку,
  - он может проделать последовательность шагов, заканчивающуюся одной из его конечных конфигураций.
- Язык, определяемый распознавателем, — это множество всех цепочек, которые допускает распознаватель.

Виды распознавателей

- По **видам считывающего устройства**
  - *двусторонние* и *односторонние*
- По **видам устройства управления**
  - детерминированные и недетерминированные
- По **видам внешней памяти**:
  - распознаватели *без внешней памяти*;
  - распознаватели *с ограниченной внешней памятью*;
  - распознаватели *с неограниченной внешней памятью*.

Классификация распознавателей по типам языков

- Для *языков с фразовой структурой* (тип 0) необходим распознаватель, равномощный машине Тьюринга
  - *недетерминированный двусторонний автомат, с неограниченной внешней памятью*. Практического применения не имеет
- Для *контекстно-зависимых языков* (тип 1)
  - *двусторонние недетерминированные автоматы с линейно ограниченной внешней памятью*. Экспоненциальная сложность
- Для *контекстно-свободных языков* (тип 2)
  - *односторонние недетерминированные автоматы с магазинной внешней памятью — МП-автоматы*. Полиномиальная сложность
  - Детерминированные КС-языки – ДМП-автоматы
- Для *регулярных языков (тип 3)*
  - *детерминированные автоматы без внешней памяти — конечные автоматы* (КА). Линейная сложность

Задача разбора

- Задача разбора в общем виде:
  - на основе имеющейся грамматики некоторого языка построить распознаватель для этого языка
- Работа распознавателей в составе компиляторов
  - сводится к построению дерева разбора входной цепочки. Затем уже это дерево разбора используется компилятором для синтеза результирующего кода
  - не только установить факт присутствия ошибки во входной программе, но и по возможности определить тип ошибки и то место в цепочке символов, где она встречается

Следующая тема:

«Регулярные выражения»