Формални Езици и Езикови Процесори
ТУ, кат. КС, летен семестър 2012

Лекция 6

Тема:

Разпознаватели
(Стекови Автомати)

Съдържание:

- КЗГ и ГФС
- Класификация на Хомски
- КА – продължение
  - От РИ към КА
  - От ЛГ към КА
  - Единство РЕ(РИ) ≡ ЛЕ(ЛГ) ≡ АЕ(КА)
- Стекови автомати
  - Въведение и Примери

До сега са въведени:

РИ

КСГ

От КСГ към друг тип граматики

- Задача: Да се съставят КСГ продукции, описващи следния език на тройни симетрични низове

$$L = \{ a^n b^n c^n \mid_{n>=1} \}$$

5 мин. За решение, ако е възможно.

Едно решение извън КСГ е показано на сл. Слайд.

От КСГ към друг тип граматики

Решението:

$S \rightarrow a\,S\,B\,C \mid a\,b\,C$

$C\,B \rightarrow B\,C$

$b\,B \rightarrow b\,b$

$b\,C \rightarrow b\,c$

$c\,C \rightarrow c\,c$

Класификация на граматиките

Хомски: формални граматики като с-ма (Σ, N, P, S):

Class 0	Граматики с фразова структура (Без ограничения в правилата) ГФС Общ формат: $\alpha \rightarrow \beta$
Class 1	Контекстнозависими граматики КЗГ Общ формат: $u1\ X\ u2 \rightarrow u1\ \alpha\ u2$
Class 2	Контекстносвободни граматики КСГ Общ формат: $X \rightarrow \alpha$
Class 3	Линейни граматики ЛГ Общ формат: $X \rightarrow a$ $X \rightarrow Y\ a$

Граматиките на Хомски

ГФС/Class 0/ представя универсалното множество на всички граматики (езици).

КЗГ/Class 1/ е подмножество на ГФС.

КСГ/Class2/ е подмножество на КЗГ /class1/.

ЛГ/Class3/ е подмножество на КСГ /class2/.

ЛГ(X $\rightarrow$ a, X $\rightarrow$ Y a) като частен случай на КСГ

- Граматиката на идентификаторите бе въведена като КСГ, но тя е ЛГ.

<Ident> ::= Letter | <Ident> Letter | <Ident> Digit

- Граматиката на целите десетични константи без знак бе въведена като КСГ, но тя е ЛГ.

<Const> ::= Digit | <Const> Digit

Релация Граматика - Език

1. Един формален език може да се опише с повече от една граматика /еквивалентни граматики/.

Пример: езикът на симетричните низове се описва с три различни граматики.

1. Всяка граматика описва само един език.

Къде са РЕ?

Отговорът:

Регулярни Езици /РИ/ ≡

Линейни Езици /ЛГ/ ≡

Автоматни Езици /КА/

Class 3 граматики на Хомски

- Припомняне:
- Как се задават /описват/ РЕ?
- Как се задават /описват/ ЛЕ?
- Как се задават /описват/ АЕ?

$$\text{РЕ} \equiv \text{ЛЕ} \equiv \text{АЕ}$$

Теореми върху РИ и АГ

- Теорема Т1:

  От Регулярен Израз (РИ) към Краен Автомат (КА).

- Теорема Т2 (обратна на Т1):

  От Краен Автомат (КА) към Регулярен Израз (РИ).

- Обобщение:

  Регулярни Езици (РИ) ≡ Автоматни Езици (КА)

Теореми върху РИ и АГ

- Теорема Т1: **От РИ към КА**

От Регулярен Израз към Краен Автомат

Даден е РИ, описващ РЕ /множество от низове/.

Т1: За всеки РЕ, описван с РИ, съществува КА разпознавател на низове от този език.

Теореми върху РИ и АГ

- Теорема Т2 (обратна на Т1): **От КА>РИ** От Краен Автомат към Регулярен Израз

Т2: Всеки Автоматен Език, (разпознаван с КА) може да бъде описан с Регулярен Израз /РИ/.

Алгоритъм: От РИ към КА

Вход: РИ или синтактичен граф генератор

Изход: Граф на преходите на КА

Алгоритъм:

1. На възел Начало се съпоставя начално състояние q_0.
2. На възел Край се съпоставя финално състояние q_f.
3. На всеки елементарен генератор се съпоставя вътрешно състояние q_i.
4. Две вътрешни състояния се свързват с дъга, ако са достижими едно от друго (има връзка между съответните елементарни генератори).
5. Дъгите се именоват със символа на генератора, към който дъгата води (влиза в съответно състояние).
6. Така конструираният граф на КА се минимизира.

Алгоритъм: От РИ към КА

Вход: РИ или синтактичен граф генератор на идентификатори

$$L\ .\ (L\ |\ D)^*$$

Изход: граф на преходите на КА, разпознавател на идентификатори

Алгоритъм: От РИ към КА

Вход: РИ или синтактичен граф генератор на цели десетични константи без знак

$$D^+$$

Изход: граф на преходите на КА, разпознавател на цели десетични константи без знак

Алгоритъм: От РИ към КА

Вход: РИ или синтактичен граф генератор

$$(a \mid b)^* c\ d$$

Изход: граф на преходите на КА, разпознавател

Алгоритъм: От РИ към КА

Вход: РИ или синтактичен граф генератор

$$a\ b\ (\ c\ d\ |\ e\ f\ )^*\ g\ h$$

Изход: граф на преходите на КА, разпознавател

Теореми върху ЛГ и АГ

- Теорема Т3:
  От Линейна Граматика (ЛГ) към Краен Автомат (КА).

- Теорема Т4 (обратна на Т3):
  От Краен Автомат (КА) към Линейна Граматика (ЛГ).

Обобщение:

Линейни Езици (ЛГ) ≡ Автоматни Езици (КА)

Теореми върху ЛГ и АГ

- Теорема Т3: **От ЛГ към КА**

  От Линейна Граматика към Краен Автомат

Дадена е ЛГ, описваща ЛЕ /множество от низове/.

Т3: За всеки ЛЕ, описван с ЛГ, съществува КА разпознавател на низове от този език.

Теореми върху ЛГ и АГ

- Теорема Т4 (обратна на Т3): **От КА>ЛГ**

От Краен Автомат към Линейна Граматика

Т4: Всеки Автоматен Език, (т.е разпознаван с КА) може да бъде описан с Линейна Граматика /ЛГ/.

Алгоритъм: От ЛГ към КА

Вход: Линейна Граматика

Изход: Граф на преходите на КА

Алгоритъм:

1. КА Азбука = ЛГ Азбука
2. КА м-во с-ния Q = N | {q0} | {qf} | {qerror}
3. На всяко правило $X \rightarrow a$, съответства преход от начално състояние q_0 в състояние X под въздействие на входен символ a
4. На всяко правило $X \rightarrow Ya$, съответства преход от състояние Y в състояние X под въздействие на входен символ a
5. Преходът от състояние S (стартов нетерминал) във финално състояние q_f се извършва под въздействие на ограничителя на входния поток eod(#).

Алгоритъм: От ЛГ към КА

Вход: Линейна Граматика на идентификатори

$$Id \rightarrow let \mid Id\ let \mid Id\ dig$$

Изход: граф на преходите на КА, разпознавател на идентификатори

Алгоритъм: От ЛГ към КА

Вход: Линейна Граматика на цели десетични константи без знак

$$C \rightarrow \text{dig} \mid C \ \text{dig}$$

Изход: граф на преходите на КА, разпознавател на цели десетични константи без знак

Алгоритъм: От ЛГ към КА

Вход: Линейна Граматика на цели десетични константи със знак

$$C \to dig \mid C\ dig$$
$$C \to S\ dig$$
$$S \to + \mid -$$

Изход: граф на преходите на КА, разпознавател на цели десетични константи със знак

Алгоритъм: От ЛГ към КА

Вход: Линейна Граматика на реални десетични константи със знак

$$C \rightarrow dig \mid C\ dig$$

$$C \rightarrow S\ dig$$

$$S \rightarrow + \mid -$$

$$F \rightarrow C\ . \mid F\ dig$$

Изход: граф на преходите на КА, разпознавател на реални десетични константи със знак

Заключение

- От теореми Т1 и Т2 следва:

  Регулярни Езици РЕ $\equiv$ Автоматни Езици АЕ

- От теореми Т3 и Т4 следва:

  Линейни Езици ЛЕ $\equiv$ Автоматни Езици АЕ

Обобщено заключение:

$$\text{РЕ} \equiv \text{ЛЕ} \equiv \text{АЕ}$$

Съответствие Граматика, Език, Разпознавател

- Разпознавател КСГ – КА с вградена стекова памет (pushdown automata).

Граматика	**Език**	**Разпознавател**
ГФС	ЕФС	Машина на Тюринг
КЗГ	КЗЕ	Линейно ограничен автомат
КСГ	КСЕ	Стеков автомат
ЛГ	ЛЕ	Краен автомат

Стеков автомат

Дефиниция на стеков автомат

Въведение в КСГ разпознаватели.

Обща схема (структура):

Входна лента (съдържа низ за сканиране);

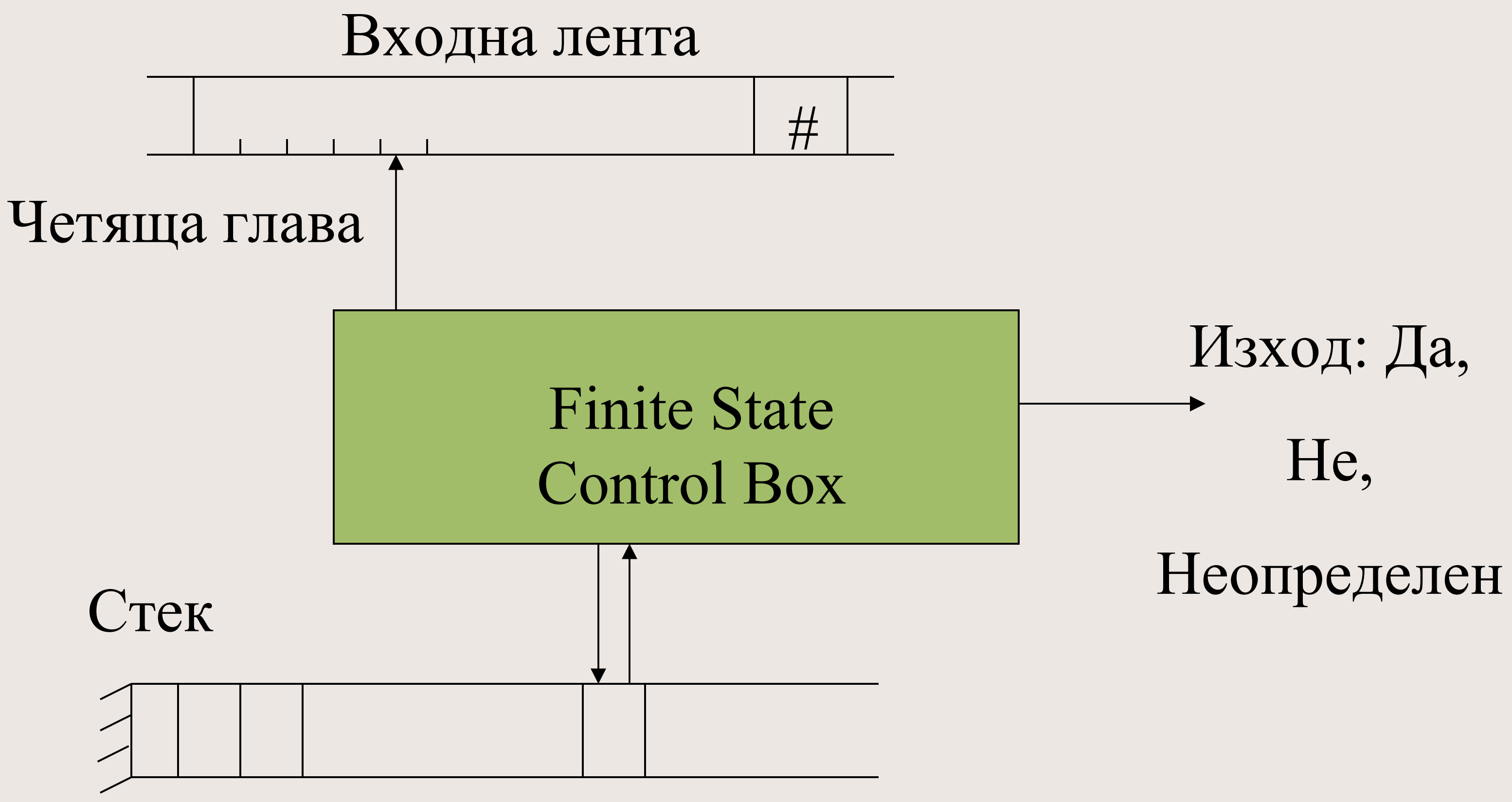
Сканиращ показалец – четяща глава;

Управляващо устройство с краен брой състояния;

Изход - /Да, Не, Неопределен/;

Вградена спомагателна стекова памет.

Стеков автомат като разпознавател



Дефиниция на стеков автомат

$PA=(\Sigma,Q,\Gamma,\delta,\lambda,P,Z_0)$	$PA=(\Sigma,Q,\delta,q_0,Q_f,P,Z_0)$
Входна азбука;	Входна азбука;
М-во вътр. състояния;	М-во вътр. състояния;
Изходна азбука;	Функция на преходите;
Функция на преходите;	Начално състояние;
Функция на изхода;	М-во закл. състояния;
Азбука на стека;	Азбука на стека;
Маркер дъно на стек;	Маркер дъно на стек;

СА като формална система:

$СА=(\Sigma, Q, \Gamma, \delta, \lambda, P, Z_0)$ $СА=(\Sigma, Q, \delta, q_0, Q_f, P, Z_0)$

Σ – входна азбука	Σ – входна азбука
Q – М-во вътр. състояния	Q – М-во вътр. състояния
Γ – азбука на изхода	δ – ф-ия на преходите: $\Sigma x Q x P \to Q x P^*$
δ – ф-ия на преходите: $\Sigma x Q x P \to Q x P^*$	q_0 – начално вътрешно състочние
λ – ф-ия на изхода: $\Sigma x Q \to \Gamma$ or $Q \to \Gamma$	Q_f – м-во закл. състояния
P – азбука на стека	P – азбука на стека
Z_0 – маркер дъно на стек	Z_0 – маркер дъно на стек

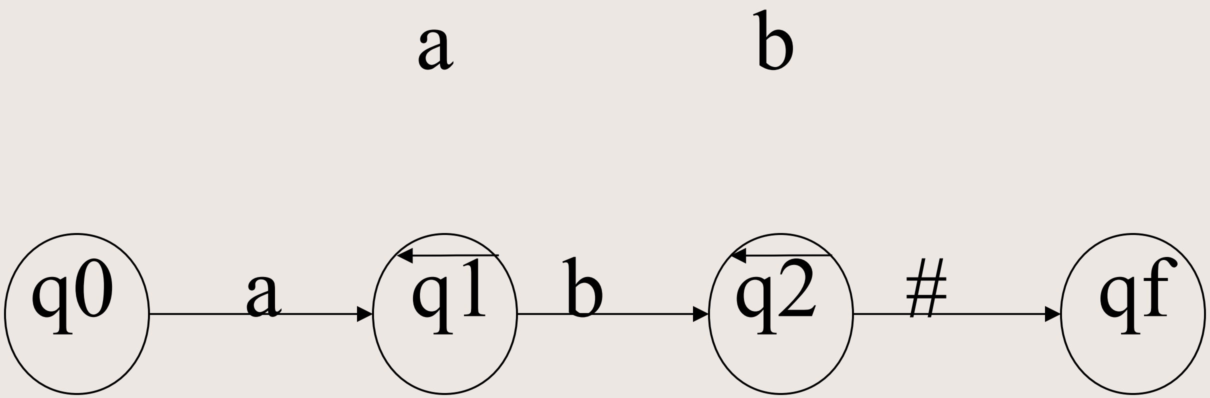
Стеков автомат - пример

- Език на симетричните низове

$$L = \{ a^n b^n \mid_{n>=1} \}$$

- Програмна реализация на стек
  - Чрез променлива брояч;
  - Чрез статична структура данни /масив/;
  - Чрез динамична структура данни;
  - Чрез инстанция /обект/ от клас стек.

Граф на преходите



Функция на преходите $Q \times \Sigma \times P \rightarrow Q \times P^*$

- $Q x \Sigma x P \qquad \rightarrow Q \times P^*$

Функция на преходите $Q \times \Sigma \times P \to Q \times P^*$

- $Q x \Sigma x P \quad \to Q \times P^*$
  - $q0, a, \nabla \to \quad q1, \nabla a$ rule 1

Функция на преходите $Q \times \Sigma \times P \to Q \times P^*$

- $Q \times \Sigma \times P \quad \to Q \times P^*$
  - $q0, a, \triangledown \quad \to q1, \triangledown a$ rule 1
  - $q1, a, a \quad \to q1, aa$ rule 2

Функция на преходите
$Q \times \Sigma \times P \rightarrow Q \times P^*$

- $Q \times \Sigma \times P \rightarrow Q \times P^*$
  - q0, a, ▽ → q1, ▽a rule 1
  - q1, a, a → q1, aa rule 2
  - q1, b, a → q2, ε rule 3

Функция на преходите $Q \times \Sigma \times P \rightarrow Q \times P^*$

- $Q \times \Sigma \times P \rightarrow Q \times P^*$
  - q0, a, ▽ → q1, ▽a — rule 1
  - q1, a, a → q1, aa — rule 2
  - q1, b, a → q2, ε — rule 3
  - q2, b, a → q2, ε — rule 4

Функция на преходите $Q \times \Sigma \times P \rightarrow Q \times P^*$

- QxΣxP → Q x P*
  - q0, a, ▽ → q1, ▽a rule 1
  - q1, a, a → q1, aa rule 2
  - q1, b, a → q2, ε rule 3
  - q2, b, a → q2, ε rule 4
  - q2, #, ▽ → qf, ▽ OK! rule 5

Сканиран низ aaabbb# - успех

Stack:

∇

- Rule 1: q0, a, $\nabla \rightarrow$ q1, ∇a — ∇a
- Rule 2: q1, a, a $\rightarrow$ q1, aa — ∇aa
- Rule 2: q1, a, a $\rightarrow$ q1, aa — ∇aaa
- Rule 3: q1, b, a $\rightarrow$ q2, ε — ∇aa
- Rule 4: q2, b, a $\rightarrow$ q2, ε — ∇a
- Rule 4: q2, b, a $\rightarrow$ q2, ε — ∇
- Rule 5: q2, #, $\nabla \rightarrow$ qf, ∇Success

String accepted

Сканиран низ abb# - грешка

Stack:

$\bigtriangledown$

$\bigtriangledown a$

$\bigtriangledown$

- Rule 1: q0, a, $\bigtriangledown \rightarrow$ q1, $\bigtriangledown a$
- Rule 3: q1, b, a $\rightarrow$ q1, ε
- Rule ?: q1, b, $\bigtriangledown \rightarrow$ No such a rule!

Error – string rejected

Сканиран низ aab# - грешка

Stack:

∇

∇a

∇aa

∇a

- Rule 1: q0, a, $\nabla \rightarrow$ q1, ∇a
- Rule 2: q1, a, a $\rightarrow$ q1, aa
- Rule 3: q1, b, a $\rightarrow$ q1, ε
- Rule ?: q1, #, a $\rightarrow$ No such a rule!

Error – string rejected

Благодаря

За

Вниманието