Конструирование Компиляторов, Теоретический минимум (2009)

1. **Определение грамматик типа 0 по Хомскому**

   Если на грамматику G = (N, T, P, S) не накладываются никакие ограничения, то её называют грамматикой типа 0, или грамматикой без ограничений.

2. **Определение грамматик типа 1 (неукорачивающих) по Хомскому**

   Если
   1. Каждое правило грамматики, кроме $S \to \varepsilon$, имеет вид $\alpha \to \beta$, $|\alpha| \le |\beta|$
   2. В том случае, когда $S \to \varepsilon \in P$, символ S не встречается в правых частях правил

   то грамматику называют грамматикой типа 1, или неукорачивающей.

3. **Определение детерминированной машины Тьюринга**

   $T_m = (Q, \Gamma, \Sigma, D, q_0, F)$, где
   Q — конечное множество состояний
   Γ — конечное множество символов (конечный алфавит)
   Σ — входной алфавит, $\Sigma \subseteq \Gamma \setminus \{b\}$ (b - пустой символ)
   D — правила перехода
   D: $(Q \setminus F) \times \Gamma \to Q \times \Gamma \times \{L, R\}$
   $q_0 \in Q$ — начальное состояние
   $F \subseteq Q$ — множество конечных состояний

4. **Определение недетерминированной машины Тьюринга**

   $T_m = (Q, \Gamma, \Sigma, D, q_0, F)$, где
   Q — конечное множество состояний
   Γ — конечное множество символов (конечный алфавит)
   Σ — входной алфавит, $\Sigma \subseteq \Gamma \setminus \{b\}$ (b - пустой символ)
   D — правила перехода
   D: $(Q \setminus F) \times \Gamma \to$ в множество подмножеств множества $Q \times \Gamma \times \{L, R\}$
   $q_0 \in Q$ — начальное состояние
   $F \subseteq Q$ — множество конечных состояний

5. **Определение конфигурации машины Тьюринга**

   Конфигурацией машины Тьюринга называется тройка (q, w, i), где
   $q \in Q$ — состояние машины Тьюринга
   $w \in \Gamma^*$ — вход, помещаемый на ленту машины Тьюринга
   $i \in Z$ — положение головки машины Тьюринга

6. **Определение языка, допускаемого машиной Тьюринга**

   Язык, допускаемый машиной Тьюринга — множество таких слов w, что, машина Тьюринга, находясь в состоянии $(q_0, w, 1)$ может достигнуть через конечное число переходов состояния $q \in F$.

7. **Соотношение между языками, порождаемыми грамматиками типа 0 и языками, допускаемыми МТ.**

   Класс языков, допускаемых машиной Тьюринга, эквивалентен классу языков, порождаемых грамматиками типа 0.

8. **Объяснить разницу между недетерминированной и детерминированной машиной Тьюринга**

   Детерминированная машина Тьюринга из данного состояния по данному символу может сделать не более одного перехода, недетерминированная же таким свойством не обладает.

9. **Определение регулярного множества**

   Регулярное множество в алфавите T определяется следующим образом:
   1. {} (пустое множество) — регулярное множество в алфавите T
   2. {a} — регулярное множество в алфавите T для каждого $a \in T$
   3. {ε} — регулярное множество в алфавите T
   4. Если P и Q — регулярные множества в алфавите T, то таковы же и множества
      a. $P \cup Q$ (объединение)
      b. PQ (конкатенация, то есть множество таких pq, что $p \in P$, $q \in Q$)
      c. P* (итерация: $P^* = \{\varepsilon\} \cup P \cup PP \cup PPP \cup \ldots$)
   5. Ничто другое не является регулярным множеством в алфавите T

10. **Определение регулярного выражения**

    Регулярное выражение и обозначаемое им регулярное множество определяются следующим образом:
    1. ∅ — обозначает множество {}
    2. ε — обозначает множество {ε}
    3. a — обозначает множество $\{a\}$
    4. Если РВ p и q обозначают множества P и Q соответственно, то:
       a. $(p|q)$ обозначает $P \cup Q$
       b. pq обозначает PQ
       c. (p^*) обозначает P^*
    5. Ничто другое не является регулярным выражением в данном алфавите

11. **Определение праволинейной грамматики**

    Если каждое правило грамматики имеет вид либо $A \to w$, либо $A \to wB$, где $w \in T^*$, $A \in N$, то ее называют праволинейной грамматикой или грамматикой типа 3 по Хомскому.

12. Определение недетерминированного конечного автомата

$M = (Q, \Sigma, D, q_0, F)$, где
Q — конечное непустое множество состояний
Σ — конечное множество допустимых входных символов (входной алфавит)
D — функция переходов, отображающая множество $Q \times (\Sigma \cup \{\varepsilon\})$ во множество подмножеств множества Q, определяющая поведение управляющего устройства
$q_0 \in Q$ — начальное состояние управляющего устройства
$F \subseteq Q$ — множество заключительных состояний

13. Определение детерминированного конечного автомата

Это НКА $M = (Q, \Sigma, D, q_0, F)$, где
Q — конечное непустое множество состояний
Σ — конечное множество допустимых символов (входной алфавит)
D — функция переходов, отображающая множество $Q \times (\Sigma \cup \{\varepsilon\})$ во множество подмножеств множества Q, определяющая поведение управляющего устройства
$q_0 \in Q$ — начальное состояние
$F \subseteq Q$ — множество конечных состояний,
для которого выполнены следующие условия:

1) $D(q,e) = \emptyset$ для любого $q \in Q$
2) $D(q,a)$ содержит не более одного элемента для любых $q \in Q$ и $a \in \Sigma$

14. Объяснить разницу между недетерминированным и детерминированным конечным автоматом

Детерминированный автомат является частным случаем недетерминированного автомата, который на каждом такте работы имеет возможность перейти не более чем в одно состояние и не может делать переходы по ε.

15. Определение конфигурации конечного автомата

Пусть $M = (Q, T, D, q_0, F)$ — НКА. Конфигурацией автомата M называется пара $(q, \omega) \in Q \times T^*$, где q — текущее состояние управляющего устройства, а ω — цепочка символов на входной ленте, состоящая из символов под головкой и всех символов справа от неё.

16. Определение языка, допускаемого конечным автоматом

Тактом автомата M называется бинарное отношение ⊢ , определенное на конфигурациях M следующим образом: если $p \in D(q,a)$, где $a \in T \cup \{\varepsilon\}$, то $(q, aw) \vdash (p, w)$ для всех $w \in T^*$.
Автомат M допускает цепочку ω, если $(q_0, \omega) \vdash^* (q, \varepsilon)$ для некоторого $q \in F$.
Языком, допускаемым автоматом M, называется множество входных цепочек, допускаемых автоматом M.
То есть: $\boldsymbol{L(M)} = \{\omega \mid \omega \in T^*$ и $(q_0, \omega) \vdash^* (q, \varepsilon)$ *для некоторого* $q \in F\}$

17. Определение ε-замыкания для подмножества состояний НКА

ε-замыкание (ε-closure) множества состояний R, $R \subseteq Q$ — множество состояний НКА, достижимых из состояний, входящих в R, посредством только переходов по ε, то есть множество $\mathbf{S} = \bigcup_{q \in R} \{p \mid (q, \varepsilon) \vdash^* (p, \varepsilon)\}$

18. Определение расширенной функции переходов для ДКА

Обозначим D^e – как расширенную функцию перехода для ДКА $M=(Q, \Sigma, D, q_0, F)$. Тогда:
1) $D^e(q, a) \equiv D(q, a)$, $a \in T$, $q \in Q$
2) $D^e(q, wa) \equiv D(D^e(q, w), a)$, $a \in T$, $w \in T^+$, $q \in Q$

19. Определение расширенной функции переходов для НКА

Обозначим D^e – как расширенную функцию перехода для НКА $M=(Q, \Sigma, D, q_0, F)$. Тогда:
1) $D^e(q, \varepsilon) = \varepsilon - closure(\{q\})$, $q \in Q$
2) $D^e(q, a) = \{p_1, p_2, \ldots, p_k\}$, $p_i \in D(q,a)$ $\forall i=1..k$, $q \in Q$, $a \in T$
3) $D^e(q, wa) = \varepsilon - closure(\bigcup_i D(pi, a))$, $p_i = D^e(q, w)$, $w \in T^+$, $q \in Q$

20. Определение функции firstpos для поддерева в дереве регулярного выражения

Функция *firstpos(n)* для каждого узла *n* узла синтаксического дерева регулярных выражений даёт множество позиций, которые соответствуют первым символам в цепочках, генерируемых подвыражением с вершиной *n*.

21. Определение функции lastpos для поддерева в дереве регулярного выражения

Функция *lastpos(n)* для каждого узла *n* узла синтаксического дерева регулярных выражений даёт множество позиций, которым соответствуют последние символы в цепочках, генерируемых подвыражениями с вершиной *n*.

22. Определение функции followpos для позиций в дереве регулярного выражения

Функция *followpos(i)* для позиции *i* есть множество позиций *j* таких, что существует некоторая строка *...cd...*, входящая в язык, описываемый регулярным выражением, такая, что позиция *i* соответствует вхождению *c*, а позиция *j* — вхождению *d*.

23. Сформулировать соотношение между регулярными множествами и языками, допускаемыми КА

1) Для всякого регулярного множества имеется КА, допускающий в точности это регулярное множество.
2) Язык, допускаемый конечным автоматом, есть регулярное множество.

24. Определение регулярной грамматики

Праволинейная грамматика G=(N,T,P,S) называется праволинейной регулярной, если:

1) каждое ее правило, кроме $S \rightarrow \varepsilon$, имеет вид либо $A \rightarrow w$, либо $A \rightarrow wB$, $w \in T$, $A,B \in N$
2) в том случае, когда $S \rightarrow \varepsilon \in P$, начальный символ S не встречается в правых частях правил.

Леволинейная грамматика G=(N,T,P,S) называется леволинейной регулярной, если:

1) каждое ее правило, кроме $S \rightarrow \varepsilon$, имеет вид либо $A \rightarrow w$, либо $A \rightarrow Bw$, $w \in T$, $A,B \in N$
2) в том случае, когда $S \rightarrow \varepsilon \in P$, начальный символ S не встречается в правых частях правил.

Регулярные грамматики — праволинейные или леволинейные регулярные грамматики.

25. Сформулировать соотношение между языками, порождаемыми праволинейными грамматиками и языками, допускаемыми КА

Для каждой праволинейной грамматики G существует конечный автомат M такой, что L(M)=L(G).

Для каждого конечного автомата M существует праволинейная грамматика G такая, что L(G)=L(M).

26. Определение эквивалентных состояний ДКА

Два различных состояния q и q' в конечном автомате M = (Q, T , F , H , Z) называются n-эквивалентными, $n \in N \cup \{0\}$, если, находясь в одном их этих состояний и получив на вход любую цепочку символов ω: $\omega \in T^*$, $|\omega| \le n$, автомат может перейти в одно и то же множество конечных состояний.

Состояния эквиваленты, если $n = \infty$.

27. Определение различимых состояний ДКА

Два различных состояния q и q' называются различимыми, если существует цепочка такая, что при ее разборе одно из этих состояний переход в конечное, а другое нет.

28. Определение контекстно-свободной грамматики без e-правил

Грамматика называется контекстно-свободной без e-правил, если:

1) Если каждое правило грамматики имеет вид $A \to \alpha$, где $A \in N$, $\alpha \in (N \cup T)^+$
2) Допускается $S \to \varepsilon$, если S не входит ни в какую правую часть

29. Определение контекстно-свободной грамматики

Если каждое правило грамматики имеет вид $A \to \alpha$, где $A \in N$, $\alpha \in (N \cup T)^*$, то ее называют контекстно-свободной.

30. Определение вывода в КС-грамматике

Определим на множестве $(N \cup T)^*$ грамматики $G = (N, T, P, S)$ бинарное отношение выводимости «⇒» следующим образом: если $\delta \to \gamma \in P$, то $\alpha\delta\beta \Rightarrow \alpha\gamma\beta$ для всех $\alpha, \beta \in (N \cup T)^*$. Если $\alpha_1 \Rightarrow \alpha_2$, то α_2 **непосредственно выводима** из α_1.
Если $\alpha \Rightarrow^k \beta$ $(k \ge 0)$, то существует последовательность шагов $\gamma_0 \Rightarrow \gamma_1 \Rightarrow \gamma_2 \Rightarrow \ldots \Rightarrow \gamma_{k-1} \Rightarrow \gamma_k$, где $\alpha = \gamma_0$ и $\beta = \gamma_k$.
Последовательность цепочек $\gamma_0, \gamma_1, \gamma_2, \ldots, \gamma_{k-1}, \gamma_k$ в этом случае называется **выводом** β из α.

31. Определение языка, порождаемого КС-грамматикой

Языком, порождаемым грамматикой $G = (N, T, P, S)$ (обозначается $L(G)$) называется множество всех цепочек терминалов, выводимых из аксиомы, то есть: $L(G) = \{w \mid w \in T^*, S \Rightarrow^+ w\}$

32. Определение сентенциальной формы

Сентенциальная форма — последовательность символов (терминалов и нетерминалов), выводимых из аксиомы.

33. Определение однозначной КС-грамматики

Упорядоченное помеченное дерево D называется деревом вывода цепочки ω КС-грамматике G=(N,T,P,S), если выполнены следующие условия:

1) корень дерева D помечен S

2) каждый лист помечен либо $a \in T$, либо ε

3) каждый внутренняя вершина помечена нетерминалом $A \in N$.

4) если X – нетерминал, которым помечена внутренняя вершина и $X_1,\ldots X_n$ – метки ее прямых потомков в указанном порядке, то $X \to X_1,\ldots X_n$ – правило из множества P.

5) Цепочка, составленная из выписанных слева направо меток листьев, равна w.

КС грамматика называется однозначной или детерминированной, если всякая выводимая терминальная цепочка имеет только одно дерево вывода.

34. Определение неоднозначной КС-грамматики

КС-грамматика G называется неоднозначной, если существует хотя бы одна цепочка $\alpha \subset L(G)$, для которой может быть построено два или более различных деревьев вывода.

35. Определение недетерминированного МП автомата

Недетерминированный автомат с магазинной памятью (МП-автомат) — семёрка $M = (Q, T, \Gamma, D, q_0, Z_0, F)$, где

1. Q — конечное множество состояний, представляющее всевозможные состояния управляющего устройства
2. T — конечный входной алфавит
3. Γ — конечный алфавит магазинных символов
4. D — отображение множества $Q \times (T \cup \{\varepsilon\}) \times \Gamma$ в множество всех конечных подмножеств $Q \times \Gamma^*$, называемое функцией переходов
5. $q_0 \in Q$ — начальное состояние управляющего устройства
6. $Z_0 \in \Gamma$ — символ, находящийся в магазине в начальный момент (начальный символ магазина)
7. $F \subseteq Q$ — множество заключительных состояний

36. Определение детерминированного МП автомата

Детерминированный автомат с магазинной памятью (МП-автомат) — семёрка $M = (Q, T, \Gamma, D, q_0, Z_0, F)$, где

1. Q — конечное множество состояний, представляющее всевозможные состояния управляющего устройства
2. T — конечный входной алфавит
3. Γ — конечный алфавит магазинных символов
4. D — отображение множества $Q \times (T \cup \{\varepsilon\}) \times \Gamma$ в множество всех конечных подмножеств $Q \times \Gamma^*$, называемое функцией переходов
5. $q_0 \in Q$ — начальное состояние управляющего устройства
6. $Z_0 \in \Gamma$ — символ, находящийся в магазине в начальный момент (начальный символ магазина)
7. $F \subseteq Q$ — множество заключительных состояний

Кроме того, должны выполняться следующие условия:

1. Множество $D(q, a, Z)$ содержит не более одного элемента для любых $q \in Q$, $a \in T \cup \{\varepsilon\}$, $Z \in \Gamma$
2. Если $D(q, \varepsilon, Z) \ne \emptyset$, то $D(q, a, Z) = \emptyset$ для всех $a \in T$

37. Определение конфигурации МП автомата

Конфигурацией автомата с магазинной памятью (МП автомата) называется тройка (q, w, u), где

$q \in Q$ — текущее состояние магазинного устройства

$w \in T^*$ — непрочитанная часть входной цепочки; первый символ цепочки w находится под входной головкой; если $w = \varepsilon$, то считается, что входная лента прочитана

$u \in \Gamma^*$ — содержимое магазина; самый левый символ цепочки u считается вершиной магазина; если $u = \varepsilon$, *то магазин считается пустым*

38. Определение языка, допускаемого МП автоматом

Цепочка w допускается МП автоматом $M = (Q, T, \Gamma, D, q_0, Z_0, F)$, если $(q_0, w, Z_0) \vdash^* (q, \varepsilon, u)$ для некоторых $q \in F$ и $u \in \Gamma^*$. Язык, допускаемый МП-автоматом M — множество всех цепочек, допускаемых автоматом M.

39. Что означает то, что недетерминированный МП автомат допускает опустошением магазина

Цепочка w допускается МП автоматом, если $(q_0, w, Z_0) \vdash^* (q, \varepsilon, \varepsilon)$ для некоторого $q \in Q$. В таком случае говорят, что автомат допускает цепочку *опустошением магазина*.

40. Соотношение, между языками, порождаемыми КС-грамматиками, и языками, допускаемыми недетерминированными МП автоматами

Язык является КС тогда и только тогда, когда он допускается магазинным автоматом.

41. Формулировка леммы о разрастании для КС-языков

Для любого контекстно-свободного языка $L\ \exists\ l: \forall\ \alpha \in L, |\alpha| \geq l, \alpha = xyz$

1. $|xy| \leq l$
2. $|y| > 0$
3. $xy^iz \in L, \forall\ i \geq 0$

42. Определение нормальной формы Хомского для КС-грамматики

Грамматика находится в нормальной форме Хомского, если правила вывода имеют вид:

- $A \to BC$; где $B, C \in N$
- $A \to a$
- $S \to \varepsilon$ (если $\varepsilon \in L$; S не входит ни в одну правую часть)

43. Определение правостороннего вывода в КС-грамматике

Вывод, в котором в любой сентенциальной форме на каждом шаге делается подстановка самого правого нетерминала, называется правосторонним.

44. Определение левостороннего вывода в КС-грамматики

Вывод, в котором в любой сентенциальной форме на каждом шаге делается подстановка самого левого нетерминала, называется левосторонним.

45. Какая грамматика называется леворекурсивной?

Грамматика называется леворекурсивной, если в ней имеется нетерминал A такой, что существует вывод $A \Rightarrow^+ Au$ для некоторой цепочки u.

46. Определение множества FIRST1

В1. $FIRST_1$ — множество всех терминальных символов, с которых может начинаться цепочка терминальных символов, выводимых из цели грамматики или ε, если $u \Rightarrow^* \varepsilon$.

В2. Множество **FIRST(A)** — множество терминальных символов, которыми начинаются цепочки, выводимые из A в грамматике G = (VT, VN, P, S), то есть, $FIRST(A) = \{a \in VT \mid A \Rightarrow a\alpha, A \in VN, \alpha \in (VT \cup VN)^*\}$

47. Определение множества FOLLOW1

Пусть A — нетерминал. Тогда **FOLLOW(A)** — множество терминалов a, которые могут появиться непосредственно справа от A в некоторой сентенциальной форме, то есть, множество терминалов a таких, что существует вывод вида $S \Rightarrow^* uAav$ для некоторых u и v.

48. Определение LL(1) Грамматики

Грамматики, для которых таблица предсказывающего анализатора не имеет неоднозначно определенных входов, называются LL(1)-грамматиками.

49. Определение LR(1) ситуации

LR(1)-ситуацией называется пара $[A \to \alpha.\beta, a]$, где $A \to \alpha\beta$ — правило грамматики, a — терминал или правый концевой маркер $. Вторая компонента ситуации называется аванцепочкой.

50. Определение LR(1) грамматики

В1. Если для КС-грамматики G функция *Action,* полученная в результате работы алгоритма построения LR(1)-таблицы, не содержит неоднозначно определенных входов, то грамматика называется LR(1)-грамматикой.

В2. Грамматика называется LR(1)-грамматикой, если из условий:

1) $S' \Rightarrow_r^* uAw \Rightarrow_r uvw$,
2) $S' \Rightarrow_r^* zBx \Rightarrow_r uvy$,
3) $FIRST(w) = FIRST(y)$

следует, что $uAy = zBx$ (т.е. $u=z$, $A=B$, $x=y$).

51. Какого типа конфликты могут появиться в канонической системе множеств LR(1) ситуаций?

Анализатор типа сдвиг-свертка при анализе некоторой цепочки может достигнуть конфигурации, в которой он, зная содержимое магазина и следующий входной символ, не может решить:

1) Делать ли сдвиг или свертку (**конфликт сдвиг/свертка**)
2) Какую из нескольких сверток применить (**конфликт свертка/свертка**)

52. Определение конфигурации LR-анализатора

Конфигурация LR анализатора — пара, первая компонента которой —содержимое стека, вторая — непросмотренный вход:

$(S_0 X_1 S_1 X_2 S_2 \ldots X_m S_m, a_i a_{i+1} \ldots a_n \$)$.

Эта конфигурация соответствует правой сентенциальной форме $X_1 X_2 \ldots X_m a_i a_{i+1} \ldots a_n$.

53. Как меняется конфигурация LR-анализатора при действии reduce?

Если выполняется *reduce A→μ*, то анализатор выполняет свертку, переходя в конфигурацию

$(S_0 X_1 S_1 X_2 S_2 \ldots X_{m-r} S_{m-r} AS, a_i a_{i+1} \ldots a_n \$)$, где $S = Goto[S_{m-r}, A]$ и r – длина μ, правой части вывода.

Анализатор сначала удаляет из магазина r символов состояния и r символов грамматики, так что на верхушке оказывается состояние S_{m-r}. Затем анализатор помещает в магазин A – левую часть правила вывода, и S – символ состояния, определяемый *Goto*$[S_{m-r}, A]$. На шаге свертки текущий входной символ не меняется. Для LR(1)-анализаторов $X_{m-r} \ldots X_m$ – последовательность символов грамматики, удаляемых из магазина, всегда соответствует μ - правой части правила вывода, по которому делается свертка.

54. Какие типы действий выполняет LR-анализатор?

Анализатор выполняет 4 вида действий:

1) Сдвиг
2) Свертка
3) Успешное завершение разбора
4) Обнаружение ошибки

55. Как меняется конфигурация LR-анализатора при действии shift?

Если выполняется *shift S* , то анализатор выполняет сдвиг, переходя в конфигурацию

$(S_0 X_1 S_1 X_2 S_2 \ldots X_m S_m a_i S, a_{i+1} \ldots a_n \$)$.

Таким образом, в магазин помещаются входной символ a_i и символ состояния S, определяемый $Action[S_m, a_i]$. Текущим входным символом становится a_{i+1}.

56. Что такое основа правой сентенциальной формы

Подцепочка сентенциальной формы, которая может быть сопоставлена правой части некоторого правила вывода, свертка по которому к левой части правила соответствует одному шагу в обращении правостороннего вывода, называется основой цепочки.

Составители и редакторы: **Кононов Алексей, Коляскина Екатерина, Кийко Александр**

группа 328.

Часть материалов взята с **http://www.esyr.us**

2009 год.