Санкт-Петербургский государственный
университет
Математико-механический факультет

В. А. Костин

МАТЕМАТИЧЕСКАЯ ЛОГИКА
(практика 2001)

Санкт-Петербург

2001

Построение выводов в исчислении высказываний

Секвенции служат для описания логических выводов

Секвенцией называется конструкция вида $\Gamma \Rightarrow A$ или $\Gamma \Rightarrow$, где Г – конечный список формул (может быть пустой), А – формула.

Интерпретация секвенции: При допущении списка формул Г имеет место формула А. Если $\Gamma \Rightarrow$, то список Г противоречив; если $\Rightarrow A$, то формула А выводима.

Исчисление секвенций.

Г – список формул А, В, С – формулы

Схемы аксиом. Секвенции вида $A \Rightarrow A$ называются аксиомами (считаем, что логических констант И, Л в формулах нет; если они нужны, то есть аксиомы $\Rightarrow$И, Л$\Rightarrow$).

Правила вывода – описывают преобразование секвенций.

1. Введение &.

$$\frac{\Gamma_1 \Rightarrow A \quad \Gamma_2 \Rightarrow B}{\Gamma_1, \Gamma_2 \Rightarrow A \& B}$$

2. **Удаление &.**

$$\frac{\Gamma \Rightarrow A \& B}{\Gamma \Rightarrow A} \qquad \frac{\Gamma \Rightarrow A \& B}{\Gamma \Rightarrow B}$$

3. **Введение $\vee$.**

$$\frac{\Gamma \Rightarrow A}{\Gamma \Rightarrow A \vee B} \qquad \frac{\Gamma \Rightarrow A}{\Gamma \Rightarrow B \vee A}$$

4. **Удаление $\vee$.**

$$\frac{\Gamma_1 \Rightarrow A \vee B \quad \Gamma_2, A \Rightarrow C \quad \Gamma_3, B \Rightarrow C}{\Gamma_1, \Gamma_2, \Gamma_3 \Rightarrow C}$$

5. Введение $\rightarrow$.

$$\frac{\Gamma, A \Rightarrow B}{\Gamma \Rightarrow A \rightarrow B}$$

6. **Удаление $\rightarrow$.**

$$\frac{\Gamma_1 \Rightarrow A \rightarrow B \quad \Gamma_2, \Rightarrow A}{\Gamma_1, \Gamma_2 \Rightarrow B}$$

7. **Введение $\neg$.**

$$\frac{\Gamma, A \Rightarrow}{\Gamma \Rightarrow \neg A}$$

8. **Удаление $\neg$.**

$$\frac{\Gamma, \neg A \Rightarrow}{\Gamma \Rightarrow A}$$

9. Сведение к противоречию.

$$\frac{\Gamma \Rightarrow A \quad \Gamma, \Rightarrow \neg A}{\Gamma \Rightarrow}$$

10. **Перестановка формул.**

$$\frac{\Gamma_1, A, B, \Gamma_2, \Rightarrow C}{\Gamma_1, B, A, \Gamma_2 \Rightarrow C}$$

11.**Правило лишней посылки.**

$$\frac{\Gamma \Rightarrow A}{\Gamma, B \Rightarrow A}$$

12. **Сокращение.**

$$\frac{\Gamma_1, A, A, \Gamma_2, \Rightarrow B}{\Gamma_1, A, \Gamma_2 \Rightarrow B}$$

Считаем, что $A \equiv B$ есть сокращение записи $(A \rightarrow B) \& (B \rightarrow A)$.

Выводом называется последовательность секвенций, каждая из которых или аксиома, или получена из некоторых предыдущих секвенций последовательности по одному из правил вывода.

Секвенция называется выводимой, если она является последней секвенцией некоторого вывода.

Приведенные правила определяют исчисление секвенций в классической логике.

ЗАДАЧИ. Вывести следующие секвенции:

1. $\Rightarrow A\to A$

   1. $A\Rightarrow A$ аксиома 2. $\Rightarrow A\to A$ 1, пр. 5

2. $A\to B,\ B\to C,\ A\Rightarrow C$

$$\dfrac{\dfrac{B\to C\Rightarrow B\to C\ \text{аксиома} \qquad \dfrac{A\to B\Rightarrow A\to B\ \text{аксиома}\quad A\Rightarrow A\ \text{аксиома}}{A\to B,A\Rightarrow B\ \text{пр. 6}}}{B\to C,A\to B,A\Rightarrow C\ \text{пр.6}}}{A\to B,B\to C,A\Rightarrow C\ \text{пр.10}}$$

Представление вывода в виде дерева

Правило вывода называется допустимым, если по всякому выводу, содержащему применение этого правила, можно построить вывод, не содержащий применение этого правила, так что у обоих выводов последние секвенции совпадают.

Теорема. Следующие правила допустимы:

10\`. **Обобщенная перестановка формул**.

$$\frac{\Gamma_1,A,B,\Gamma_2,\Rightarrow}{\Gamma_1,B,A,\Gamma_2\Rightarrow}$$

11\`. **Обобщенное правило лишней посылки.**

$$\frac{\Gamma\Rightarrow}{\Gamma,B\Rightarrow}$$

12\`. **Обобщенное сокращение.**

$$\frac{\Gamma_1,A,A,\Gamma_2,\Rightarrow}{\Gamma_1,A,\Gamma_2\Rightarrow}$$

Докажем допустимость правила 10\`.

Секвенция $\Gamma_1,A,B,\Gamma_2,\Rightarrow$ может быть выведена из аксиом применением на последнем шаге только правила 9, т. е.

1. $\Gamma_1,A,B,\Gamma_2,\Rightarrow C$ 2. $\Gamma_1,A,B,\Gamma_2,\Rightarrow\neg C$
3. $\Gamma_1,B,A,\Gamma_2,\Rightarrow C$ 1, пр 10 4. $\Gamma_1,B,A,\Gamma_2,\Rightarrow\neg C$ 2, пр 10
5. $\Gamma_1,B,A,\Gamma_2,\Rightarrow$ 3, 4, пр 9

Аналогично доказывается допустимость правил 11\` и 12\`.

3. $\Rightarrow\neg\neg A\equiv A$ т.е. $(\neg\neg A\to A)\&(A\to\neg\neg A)$

   1. $\neg\neg A\Rightarrow\neg\neg A$ аксиома 2. $\neg A\Rightarrow\neg A$ аксиома
   3. $\neg A,\neg\neg A\Rightarrow$ 2, 1, пр. 9 4. $\neg\neg A,\neg A\Rightarrow$ 3, пр. 10\`

5. $\neg\neg A \Rightarrow A$ 4, пр. 8
6. $\Rightarrow \neg\neg A \to A$ 5, пр. 5
7. $A \Rightarrow A$ аксиома
8. $A, \neg A \Rightarrow$ 7, 2, пр. 9
9. $A \Rightarrow \neg\neg A$ 8, пр. 7
10. $\Rightarrow A \to \neg\neg A$ 9, пр. 5
11. $\Rightarrow (\neg\neg A \to A)\&(A \to \neg\neg A)$ 6, 10, пр. 1.

4. $A \to (B \to C), A \to B, A \Rightarrow C$

1. $A \to B \Rightarrow A \to B$ аксиома
2. $A \Rightarrow A$ аксиома
3. $A \to (B \to C) \Rightarrow A \to (B \to C)$ аксиома
4. $A \to B, A \Rightarrow B$ 1, 2, пр. 6
5. $A \to (B \to C), A \Rightarrow B \to C$ 3, 2, пр. 6
6. $A \to (B \to C), A, A \to B, A \Rightarrow C$ 5, 4, пр. 6
7. $A \to (B \to C), A \to B, A, A \Rightarrow C$ 6, пр. 10
8. $A \to (B \to C), A \to B, A \Rightarrow C$ 7, пр. 12

5. $A \to B, \neg B \Rightarrow \neg A$

1. $A \to B \Rightarrow A \to B$ аксиома
2. $A \Rightarrow A$ аксиома
3. $\neg B \Rightarrow \neg B$ аксиома
4. $A \to B, A \Rightarrow B$ 1, 2, пр. 6
5. $A \to B, A, \neg B \Rightarrow$ 4, 3, пр. 9
6. $A \to B, \neg B, A \Rightarrow$ 5, пр. 10
7. $A \to B, \neg B \Rightarrow \neg A$ 6, пр. 7

6. $A, \neg B \Rightarrow \neg(A \to B)$

1. $A \to B \Rightarrow A \to B$ аксиома
2. $A \Rightarrow A$ аксиома
3. $\neg B \Rightarrow \neg B$ аксиома
4. $A \to B, A \Rightarrow B$ 1, 2, пр. 6
5. $A \to B, A, \neg B \Rightarrow$ 3, 4 пр. 9
6. $A, \neg B, A \to B \Rightarrow$ 5, пр. 10, пр. 10
7. .$A, \neg B \Rightarrow \neg(A \to B)$ 6, пр.7

7. $A \to B, B \to C \Rightarrow A \to C$

1. $A \to B, B \to C, A \Rightarrow C$ упр. 2
2. $A \to B, B \to C \Rightarrow A \to C$ 1, пр 6

Теорема. Следующие правила допустимы:

а. **Сечение.**

$$\frac{\Gamma_1 \Rightarrow A \qquad \Gamma_2, A \Rightarrow B}{\Gamma_1, \Gamma_2 \Rightarrow B}$$

доказательство

1. $\Gamma_1 \Rightarrow A$
2. $\Gamma_2, A \Rightarrow B$
3. $\Gamma_2, \Rightarrow A \to B$ 2, пр. 5
4. $\Gamma_2, \Gamma_1 \Rightarrow B$ 3, 4, пр.6
5. $\Gamma_1, \Gamma_2 \Rightarrow B$ 5, пр.10,…пр. 10

б. **Объединение посылок.**

$$\frac{\Gamma, A, B \Rightarrow C}{\Gamma, A\&B \Rightarrow C}$$

доказательство

1. $\Gamma, A, B \Rightarrow C$
2. $A\&B \Rightarrow A\&B$ аксиома
3. $A\&B \Rightarrow A$ 2, пр.2_1
4. $A\&B \Rightarrow B$ 2, пр.2_2
5. $A\&B, \Gamma, A \Rightarrow C$ 1,4 пр. а
6. $A\&B, A\&B, \Gamma \Rightarrow C$ 5,3 пр. а
7. $A\&B, \Gamma \Rightarrow C$ 6, пр.12
8. $\Gamma, A\&B \Rightarrow C$ 7, пр.10,…пр. 10

в. **Расщепление посылок**.

$$\frac{\Gamma, A\&B \Rightarrow C}{\Gamma, A, B \Rightarrow C}$$

доказательство

1. $\Gamma, A\&B \Rightarrow C$
2. $A \Rightarrow A$ аксиома
3. $B \Rightarrow B$ аксиома
4. $A, B, \Gamma, A\&B \Rightarrow C$
5. $A, B, \Gamma \Rightarrow C$ 4,1, пр. а
6. $\Gamma, A, B \Rightarrow C$ 5, пр.10,…пр. 10

г**. Разбор случаев**.

$$\frac{\Gamma, A \Rightarrow C \qquad \Gamma, B \Rightarrow C}{}$$

доказательство

1. $\Gamma, A \Rightarrow C$
2. $\Gamma, B \Rightarrow C$
3. $A \vee B \Rightarrow A \vee B$ аксиома
4. $A \vee B, \Gamma, \Gamma \Rightarrow C$ 3,2,1, пр.4
5. $\Gamma, A \vee B \Rightarrow C$ 4, пр. 10,…,пр.12,…,пр10

$\Gamma, A \vee B \Rightarrow C$

д. **Контрапозиция.**

$$\frac{\Gamma, A \Rightarrow B}{\Gamma, \neg B \Rightarrow \neg A}$$

доказательство

1.$\Gamma, A, \Rightarrow B$ 2. $\Gamma, A, \neg B \Rightarrow B$ 1, пр. 11

3 $\Gamma, \neg B, A \Rightarrow B$ 2, пр. 10. 4.$\neg B \Rightarrow \neg B$ аксиома

5. $\Gamma, \neg B, A \Rightarrow \neg B$ 4, пр.11,…,пр. 11,…,пр. 10,…,пр. 10

6. $\Gamma, \neg B, A \Rightarrow$ 3,5, пр.9 7.$\Gamma, \neg B \Rightarrow \neg A$ 6, пр.7

е. **Доказательство от противного.**

$$\frac{\Gamma, \neg A \Rightarrow \neg B}{\Gamma, B \Rightarrow A}$$

доказательство

1.$\Gamma, \neg A, \Rightarrow \neg B$ 2. $\Gamma, \neg A, B \Rightarrow \neg B$ 1, пр. 11

3 $\Gamma, B, \neg A \Rightarrow \neg B$ 2, пр. 10. 4.$B \Rightarrow B$ аксиома

5. $\Gamma, B \neg, A \Rightarrow B$ 4, пр.11,…,пр. 11,…,пр. 10,…,пр. 10

6. $\Gamma, B, \neg A \Rightarrow$ 3,5, пр.9 7.$\Gamma, \neg B \Rightarrow \neg A$ 6, пр.7

ж. **Введение & и →.**

$$\frac{A_1, \ldots, A_k \Rightarrow B}{\Rightarrow A_1 \& \ldots \& A_k \to B}$$

доказательство

1. $A_1, \ldots, A_k \Rightarrow B$ 1, пр. б k-1 раз
2. $\Rightarrow A_1 \& \ldots \& A_k \to B$

з. **Удаление & и →.**

$$\frac{\Rightarrow A_1 \& \ldots \& A_k \to B}{A_1, \ldots, A_k \Rightarrow B}$$

доказательство

1. $\Rightarrow A_1 \& \ldots \& A_k \to B$
2. $A_1 \& \ldots \& A_k \Rightarrow A_1 \& \ldots \& A_k$ аксиома
3. $A_1 \& \ldots \& A_k \Rightarrow B$ 2,1, пр. а

.4.$A_1, \ldots, A_k \Rightarrow B$ 4, пр. в k-1 раз

8. $A \to (B \to C) \Rightarrow B \to (A \to C)$

1. $A \Rightarrow A$ аксиома 2. $A \to (B \to C) \Rightarrow A \to (B \to C)$ аксиома

3. $A \to (B \to C), A \Rightarrow B \to C$ 2, 1, пр. 6 4. $B \Rightarrow B$ аксиома

5. $A \to (B \to C), A, B \Rightarrow C$ 3, 4, пр. 6 6. $A \to (B \to C), B, A \Rightarrow C$ 5, пр. 10

7. $A \to (B \to C), B \Rightarrow A \to C$ 6, пр. 5 8. $A \to (B \to C) \Rightarrow B \to (A \to C)$ 7 пр.5

9. $A \to (B \to C) \Rightarrow A \& B \to C$

1.$A \to (B \to C), A, B \Rightarrow C$ упр. 8.5 2.$A \to (B \to C), A, B \Rightarrow C$ 1, пр. б

3.$A \to (B \to C) \Rightarrow A \& B \to C$ 2, пр. 5

10. $A \& B \to C \Rightarrow A \to (B \to C)$

1.$A \& B \to C \Rightarrow A \& B \to C$ аксиома 2.$A \& B \Rightarrow A \& B$ аксиома

3.$A \& B \to C, A \& B \Rightarrow C$ 1, 2, пр. 6 4.$A \& B \to C, A, B \Rightarrow C$ 3, пр. б

5$A \& B \to C, A \Rightarrow B \to C$ 4, пр. 5 6.$A \& B \to C, \Rightarrow A \to (B \to C)$ 5, пр. 5

11.$A \to B \Rightarrow (B \to C) \to (A \to C)$

1.$A \to B \Rightarrow A \to B$ аксиома 2.$A \Rightarrow A$ аксиома

3.$A \to B, A \Rightarrow B$ 1, 2, пр. 6 4.$B \to C \Rightarrow B \to C$ аксиома

5.$B \Rightarrow B$ аксиома 6.$B \to C, B \Rightarrow C$ 4, 5, пр.6

7.$A \to B, A, B \to C \Rightarrow C$ 3, 6, пр. а 8.$A \to B, B \to C, A \Rightarrow C$ 7, пр. 10

9.$A \to B, B \to C \Rightarrow A \to C$ 8, пр. 5 10.$A \to B \Rightarrow (B \to C) \to (A \to C)$ 9, пр. 5

12. $A \to B \Rightarrow (C \to A) \to (C \to B)$

1. $A \to B \Rightarrow A \to B$ аксиома 2. $A \Rightarrow A$ аксиома

3. $A \to B, A \Rightarrow B$ 1, 2 пр. 6 4. $C \to A \Rightarrow C \to A$ аксиома

5. $C \Rightarrow C$ аксиома
6. $C \to A, C \Rightarrow A$ 4, 5 пр.6
7. $C \to A, C, A \to B \Rightarrow B$ 3, 6 пр. а
8. $A \to B, C \to A, C \Rightarrow B$ 7, пр. 10, пр. 10
9. $A \to B, C \to A \Rightarrow C \to B$ 8, пр. 5
10. $A \to B \Rightarrow (C \to A) \to (C \to B)$ 9, пр. 5

13. $A \to B \Rightarrow (C \& A) \to (C \& B)$

1. $A \to B \Rightarrow A \to B$ аксиома
2. $A \Rightarrow A$ аксиома
3. $C \& B \Rightarrow C \& B$ аксиома
4. $C, B \Rightarrow C \& B$ 3, пр.б
5. $A \to B, A \Rightarrow B$ 1, 2, пр. 6
6. $A \to B, A, C \Rightarrow C \& B$ 5, 4, пр. а
7. $A \to B, C, A \Rightarrow C \& B$ 6, пр. 10
8. $A \to B, C \& A \Rightarrow C \& B$ 7, пр. б
9. $A \to B \Rightarrow (C \& A) \to (C \& B)$ 8, пр. 5

14. $A \to B \Rightarrow (A \& C) \to (B \& C)$ аналогично 13 3. $B \& C \Rightarrow B \& C$

15. $A \to B \Rightarrow (A \vee C) \to (B \vee C)$

1. $A \to B \Rightarrow A \to B$ аксиома
2. $A \Rightarrow A$ аксиома
3. $A \to B, A \Rightarrow B$ 1, 2, пр. 6
4. $A \to B, A \Rightarrow B \vee C$ 3, пр.3_1
5. $C \Rightarrow C$ аксиома
6. $C \Rightarrow B \vee C$ 5, пр.3_2
7. $C, A \vee C \Rightarrow B \vee C$ 6, пр. 11
8. $A \vee C, C \Rightarrow B \vee C$ 7, пр. 10
9. $A \vee C \Rightarrow A \vee C$ аксиома
10. $A \vee C, A \to B, A \vee C \Rightarrow B \vee C$ 9, 4, 8, пр.4
11 $A \to B, A \vee C \Rightarrow B \vee C$ 10, пр. 10, пр. 12
12. $A \to B \Rightarrow (A \vee C) \to (B \vee C)$ 11, пр. 5

16. $A \to B \Rightarrow (C \vee A) \to (C \vee B)$

1. $A \to B \Rightarrow A \to B$ аксиома
2. $A \Rightarrow A$ аксиома
3. $A \to B, A \Rightarrow B$ 1, 2, пр. 6
4. $A \to B, A \Rightarrow C \vee B$ 3, пр.3_2
5. $C \Rightarrow C$ аксиома
6. $C \Rightarrow C \vee B$ 5, пр.3_1
7. $C, C \vee B \Rightarrow C \vee B$ 6, пр. 11
8. $C \vee A, C \Rightarrow C \vee B$ 7, пр. 10
9. $C \vee A \Rightarrow C \vee A$ аксиома
10. $C \vee A, A \to B, C \vee A \Rightarrow C \vee B$ 9, 4, 8, пр.4
11 $A \to B, C \vee A \Rightarrow C \vee B$ 10, пр. 10, пр. 12
12. $A \to B \Rightarrow (C \vee A) \to (C \vee B)$ 11, пр. 5

17. $\neg A \Rightarrow A \to B$

1. $A \Rightarrow A$ аксиома
2. $\neg A \Rightarrow \neg A$ аксиома
3. $A, \neg A \Rightarrow$ 1,2, пр. 9
4. $A, \neg A, B \Rightarrow$ 3, пр. $11_{\text{обобщ}}$
5. $A, \neg A \Rightarrow B$ 4, пр.8
6. $\neg A, A \Rightarrow B$ 5, пр.10
7. $\neg A \Rightarrow A \to B$ 6, пр. 5

18. $A \Rightarrow \neg A \to B$ аналогично 17

19. $B \Rightarrow A \to B$

1. $B \Rightarrow B$ аксиома
2. $B, A \Rightarrow B$ 1, пр. 11
3. $B \Rightarrow A \to B$ 1, пр. 5

20. $A \to B \Rightarrow \neg B \to \neg A$

1. $A \to B \Rightarrow A \to B$ аксиома
2. $A \Rightarrow A$ аксиома
3. $A \to B, A \Rightarrow B$ 1, 2, пр. 6
4. $\neg B \Rightarrow \neg B$ аксиома
5. $A \to B, A, \neg B \Rightarrow$ 3,4, пр. 9
6. $A \to B, \neg B, A \Rightarrow$ 5, пр. $10_{\text{обобщ}}$
7. $A \to B, \neg B \Rightarrow \neg A$ 6, пр. 7
8. $A \to B \Rightarrow \neg B \to \neg A$ 7, пр. 5

21. $A \to \neg B \Rightarrow B \to \neg A$

1. $A \to \neg B \Rightarrow A \to \neg B$ аксиома
2. $A \Rightarrow A$ аксиома
3. $A \to B, A \Rightarrow B$ 1, 2, пр. 6
4. $B \Rightarrow B$ аксиома

5. $A\to\neg B, A, B\Rightarrow$ 3, 4, пр. 9   6. $A\to\neg B, B, A\Rightarrow$ 5, пр. $10_{\text{обобщ}}$

7. $A\to\neg B, B\Rightarrow\neg A$ 6, пр. 7   8. $A\to\neg B\Rightarrow B\to\neg A$ 7, пр. 5

22. $\neg A\to B\Rightarrow\neg B\to A$ аналогично 21 с аксиомами $\neg A\Rightarrow\neg A$, $\neg A\to B\Rightarrow\neg A\Rightarrow B$, $B\Rightarrow B$

23. $\neg A\to\neg B\Rightarrow B\to A$

1. $\neg A\to\neg B\Rightarrow\neg A\to\neg B$ аксиома   2. $\neg A\Rightarrow\neg A$ аксиома

3. $\neg A\to\neg B, \neg A\Rightarrow\neg B$ 1, 2, пр. 6   4. $B\Rightarrow B$ аксиома

5. $\neg A\to\neg B,\neg A, B\Rightarrow$ 3, 4, пр. 9   6. $\neg A\to\neg B,B,\neg A\Rightarrow$ 5, пр. $10_{\text{обобщ}}$

7. $\neg A\to\neg B,B\Rightarrow A$ 6, пр. 8   8. $\neg A\to\neg B\Rightarrow B\to A$ 7, пр. 5

Теорема 1. Пусть S_C^P - оператор подстановки вместо атома *P* формулы *C*, тогда, если выводима секвенция $A_1,\ldots,A_k\Rightarrow B$, то выводима также секвенция $S_C^P(A_1,\ldots A_k\Rightarrow B)$.

Доказывается индукцией по длине вывода секвенции $A_1,\ldots,A_k\Rightarrow B$, применяя S_C^P к каждой формуле, входящей в секвенции вывода.

Теорема 2. Доказать, что следующие правила допустимы.

И). **Введение ≡.**

$$\frac{\Gamma,A\Rightarrow B\quad \Gamma,B\Rightarrow A}{\Gamma\Rightarrow A\equiv B}$$

доказательство

1. $\Gamma,A\Rightarrow B$ 2. $\Gamma,B\Rightarrow A$

3. $\Gamma\Rightarrow A\to B$ 1, пр. 5   4. $\Gamma\Rightarrow A\to B$ 1, пр. 5

5. $\Gamma\Rightarrow(A\to B)\&(B\to A)$ 3, 4, пр. 1

К). **Замена ≡ на →.**

$$1.\ \frac{\Gamma\Rightarrow A\equiv B}{\Gamma\Rightarrow A\to B}\qquad 2.\ \frac{\Gamma\Rightarrow A\equiv B}{\Gamma\Rightarrow B\to A}$$

доказательство

1. $\Gamma\Rightarrow A\equiv B$ 2. $A\Rightarrow A$ аксиома

3. $B\Rightarrow B$ аксиома   4. $\Gamma\Rightarrow A\to B$ 1, пр. 2_1

5. $\Gamma\Rightarrow B\to A$ 1, пр. 2_2 6. $\Gamma,A\Rightarrow B$ 4, 2, пр 6

7. $\Gamma,B\Rightarrow A$ 5,3, пр 6

24. $A\to B, B\to A\Rightarrow A\equiv B$

1. $A\to B\Rightarrow A\to B$ аксиома   2. $B\to A\Rightarrow B\to A$ аксиома

3. $A\to B, B\to A\Rightarrow(A\to B)\&(B\to A)$ 1, 2, пр. 1

25. $A\equiv B\Rightarrow A\to B$

1. $A\equiv B\Rightarrow A\equiv B$ аксиома   2. $A\equiv B\Rightarrow A\to B$ 1, пр. 2

26. $A\equiv B\Rightarrow B\to A$ аналогично 25

27. $A\equiv B, A\Rightarrow B$

1. $A\equiv B\Rightarrow A\to B$ упр. 25   2. $A\to A$ аксиома

3. $A\equiv B, A\Rightarrow B$ 1, 2, пр. 6

28. $\Rightarrow A\equiv A$

1. $A\Rightarrow A$ аксиома   2. $\Rightarrow A\equiv A$ 1, 1, пр. и

29. $A\equiv B, B\equiv C\Rightarrow A\equiv C$

1. $A\to B, B\to C\Rightarrow A\to C$ упр. 7
2. $C\to B, B\to A\Rightarrow C\to A$ упр. 7
3. $A\to B, B\to C, C\to B, B\to A\Rightarrow C\equiv A$ 1, 2, пр.1
4. $B\to C,C\to B, A\to B, B\to A\Rightarrow C\equiv A$ 3, пр.10, пр.10
5. $A\equiv B, B\equiv C\Rightarrow A\equiv C$ 3,4, пр.б, пр.10, пр.10, пр.б

30. $A\equiv B\Rightarrow B\equiv A$

1. $A\equiv B\Rightarrow A\equiv B$ аксиома
2. $A\equiv B, A\Rightarrow B$ 1, пр. $к_1$
3. $A\equiv B, B\Rightarrow A$ 1, пр. $к_2$
4. $A\equiv B\Rightarrow B\equiv A$ 3, 2, пр. и

31. $A\equiv B\Rightarrow\neg A\equiv\neg B$

1. $A\to B\Rightarrow\neg B\to\neg A$ упр. 20
2. $B\to A\Rightarrow\neg A\to\neg B$ упр. 20
3. $B\to A, A\to B\Rightarrow\neg A\equiv\neg B$ 2, 1, пр. 1
4. $A\to B, B\to A\Rightarrow\neg A\equiv\neg B$ 2, 1, пр. 1
5. $A\equiv B\Rightarrow\neg A\equiv\neg B$ 4, пр. б

32. $A\equiv B\Rightarrow A\&C\equiv B\&C$

1. $A\equiv B\Rightarrow A\equiv B$ аксиома
2. $A\equiv B, A\Rightarrow B$ 1, пр. $к_1$
3. $A\equiv B, B\Rightarrow A$ 1, пр. $к_2$
4. $C\Rightarrow C$ аксиома
5. $A\equiv B, A, C\Rightarrow B\&C$ 2, 4, пр. 1
6. $A\equiv B, B, C\Rightarrow A\&C$ 3, 4, пр. 1
7. $A\equiv B, A\&C\Rightarrow B\&C$ 5, пр. Б
8. $A\equiv B, B\&C\Rightarrow A\&C$ 6, пр. б
9. $A\equiv B\Rightarrow A\&C\equiv B\&C$ 7, 8, пр. и

33. $A\equiv B\Rightarrow C\&A\equiv C\&B$ аналогично 32

34. $A\equiv B\Rightarrow A\vee C\equiv B\vee C$

1. $A\equiv B, A\Rightarrow B$ упр. 32.2
2. $A\equiv B, B\Rightarrow A$ упр. 32.3
3. $A\equiv B, A\Rightarrow B\vee C$ 1, пр. 3_1
4. $A\equiv B, B\Rightarrow A\vee C$ 2, пр. 3_2
5. $C\Rightarrow C$ аксиома
6. $B\vee C\Rightarrow B\vee C$ аксиома
7. $A\vee C\Rightarrow A\vee C$ аксиома
8. $C\Rightarrow A\vee C$ 5, пр. 3_2
9. $C\Rightarrow B\vee C$ 5, пр. 3_2
10. $B\vee C, A\equiv B\Rightarrow A\vee C$ 6, 4, 8, пр. 4
11. $A\vee C, A\equiv B\Rightarrow B\vee C$ 7, 3, 9, пр. 4
12. $A\equiv B, A\vee C\Rightarrow A\vee C$ 10, пр. 10
13. $A\equiv B, A\vee C\Rightarrow B\vee C$ 11, пр. 10
14. $A\equiv B\Rightarrow A\vee C\equiv B\vee C$ 13, 14, пр. и

35. $A\equiv B\Rightarrow C\vee A\equiv C\vee B$ аналогично 34

36. $A\equiv B\Rightarrow A\to C\equiv B\to C$

1. $A\equiv B\Rightarrow A\to B$ упр.25
2. $A\to B\Rightarrow(B\to C)\to(A\to C)$ упр.10
3. $A\equiv B\Rightarrow(B\to C)\to(A\to C)$ 1, 2, пр. а
4. $B\to C\Rightarrow B\to C$ аксиома
5. $A\equiv B, B\to C\Rightarrow A\to C$ 3, 4, пр. 6
6. $A\equiv B\Rightarrow B\to A$ упр.26
7. $B\to A\Rightarrow(A\to C)\to(B\to C)$ упр.10
8. $A\to C\Rightarrow A\to C$ аксиома
9. $A\equiv B\Rightarrow(A\to C)\to(B\to C)$ 6, 7, пр. А
10. $A\equiv B, A\to C\Rightarrow B\to C$ 9, 8, пр. 6
11. $A\equiv B\Rightarrow A\to C\equiv B\to C$ 10, 5, пр. и

37. $A\equiv B\Rightarrow C\to A\equiv C\to B$ аналогично 36 с использованием упр. 12

Теорема (о замене). Пусть A – формула, B – подформула A; A_1 результат замены в A некоторого вхождения B на формулу B_1, тогда выводима секвенция $B\equiv B_1\Rightarrow A\equiv A_1$.

Доказывается индукцией по построению формулы A из B, используя секвенции 25÷37.

38. $\Rightarrow A\&B\equiv B\&A$

1. $A\&B\Rightarrow A\&B$ аксиома
2. $A\&B\Rightarrow A$ 1, пр. 2_1
3. $A\&B\Rightarrow B$ 1, пр. 2_2
4. $A\&B, A\&B\Rightarrow B\&A$ 3, 2, пр. 1
5. $A\&B\Rightarrow B\&A$ 4, пр. 12
6. $B\&A\Rightarrow B\&A$ аксиома
7. $B\&A\Rightarrow A$ 6, пр. 2_2
8. $B\&A\Rightarrow B$ 7, пр. 2_1
9. $B\&A, B\&A\Rightarrow A\&B$ 9, 8, пр. 1
10. $B\&A\Rightarrow A\&B$ 4, пр. 12

11. $A\&B \equiv B\&A$ 5,10, пр. и

39. $\Rightarrow A\vee B \equiv B\vee A$

1. $A\vee B \Rightarrow A\vee B$ аксиома
2. $A \Rightarrow A$ аксиома
3. $B \Rightarrow B$ аксиома
4. $A \Rightarrow B\vee A$ 2, пр.3_1
5. $B \Rightarrow B\vee A$ 3, пр.3_2
6. $A\vee B \Rightarrow B\vee A$ 1, 4, 5, пр.4
7. $B\vee A \Rightarrow B\vee A$ аксиома
8. $A \Rightarrow A\vee B$ 2, пр. 3_1
9. $B \Rightarrow A\vee B$ 2, пр. 3_2
10. $A\vee B \Rightarrow B\vee A$ 7, 8, 9, пр.4
11. $A\vee B \equiv B\vee A$ 6, 10, пр. и

40. $\Rightarrow A\&(B\&C) \equiv (A\&B)\&C$

1. $(A\&B)\&C \Rightarrow (A\&B)\&C$ аксиома
2. $A, B, C \Rightarrow (A\&B)\&C$ 1, пр. в, пр.10, пр. в, пр.10, пр.10
3. $A, B\&C \Rightarrow (A\&B)\&C$ 2, пр. б
4. $A\&(B\&C) \Rightarrow (A\&B)\&C$ 2, пр. б
5. $A\&(B\&C) \Rightarrow A\&(B\&C)$ аксиома
6. $C, A, B \Rightarrow A\&(B\&C)$ 5 ,пр. в, пр.в, пр. в, пр.10, пр.10
7. $A\&B, C \Rightarrow A\&(B\&C)$ 6, пр. б, пр.10,
8. $(A\&B)\&C \Rightarrow A\&(B\&C)$ 5, пр. в, пр. в, пр. в, пр.10,
9. $\Rightarrow A\&(B\&C) \equiv (A\&B)\&C$ 4, 8, пр. и

41. $\Rightarrow A\vee(B\vee C) \equiv (A\vee B)\vee C$ аналогично 40, используя пр. г

42. $\Rightarrow A\&(B\vee C) \equiv (A\&B)\vee(A\&C)$

1. $A\&B \Rightarrow A\&B$ аксиома
2. $A\&C \Rightarrow A\&C$ аксиома
3. $A, B \Rightarrow A\&B$ 1, пр. в
4. $A, C \Rightarrow A\&C$ 2, пр. в
5. $A, B \Rightarrow (A\&B)\vee(A\&C)$ 3, пр. 3_1
6. $A, C \Rightarrow (A\&B)\vee(A\&C)$ 3, пр. 3_2
7. $A, B\vee C \Rightarrow (A\&B)\vee(A\&C)$ 5, 6, пр. г
8. $A\&(B\vee C) \Rightarrow (A\&B)\vee(A\&C)$ 7, пр. б
9. $(A\&B)\vee(A\&C) \Rightarrow (A\&B)\vee(A\&C)$ аксиома
10. $A\&B \Rightarrow A$ 1, пр. 2_1
11. $A\&C \Rightarrow A$ 1, пр. 2_1
12. $A\&B \Rightarrow A\vee(B\&C)$ 10, пр. 3_1
13. $A\&C \Rightarrow A\vee(B\&C)$ 11, пр. 3_1
14. $(A\&B)\vee(A\&C) \Rightarrow A\vee(B\&C)$ 9, 12, 13, пр. 4
15. $\Rightarrow A\&(B\vee C) \equiv (A\&B)\vee(A\&C)$ 8, 14, пр. и

43. $\Rightarrow A\vee(B\&C) \equiv (A\vee B)\&(A\vee C)$

1. $A \Rightarrow A$ аксиома
2. $A \Rightarrow A\vee B$ 1, пр. 3_1
3. $A \Rightarrow A\vee C$ 1, пр. 3_1
4. $A, A \Rightarrow (A\vee B)\&(A\vee C)$ 2, 3, пр. 1
5. $A \Rightarrow (A\vee B)\&(A\vee C)$ 4, пр. 12
6. $B\&C \Rightarrow B\&C$ аксиома
7. $B\&C \Rightarrow B$ 6, пр.2_1
8. $B\&C \Rightarrow C$ 6, пр.2_2
9. $B\&C \Rightarrow A\vee B$ 7, пр.3_2
10. $B\&C \Rightarrow A\vee C$ 8, пр.3_2
11. $B\&C, B\&C \Rightarrow (A\vee B)\&(A\vee C)$ 9, 10, пр.1
12. $B\&C \Rightarrow (A\vee B)\&(A\vee C)$ 11, пр.12
13. $A\vee(B\&C) \Rightarrow (A\vee B)\&(A\vee C)$ 5, 12, пр. г
14. $A \Rightarrow A\vee(B\&C)$ 1, пр. 3_1
15. $B\&C \Rightarrow A\vee(B\&C)$ 6, пр. 3_2
16. $B, C \Rightarrow A\vee(B\&C)$ 15, пр. в
17. $B, A \Rightarrow A\vee(B\&C)$ 14, пр. 11, пр. 10
18. $B, A\vee C \Rightarrow A\vee(B\&C)$ 16, 17, пр. 11
19. $A\vee C, A \Rightarrow A\vee(B\&C)$ 14, пр. 11, пр. 10
20. $A\vee C, B \Rightarrow A\vee(B\&C)$ 18, пр. 10
21. $A\vee C, A\vee B \Rightarrow A\vee(B\&C)$ 19, 20, пр. г
22. $A\vee B, A\vee C \Rightarrow A\vee(B\&C)$ 21, пр. 10
23. $(A\vee B)\&(A\vee C) \Rightarrow A\vee(B\&C)$ 22, пр. б
24. $\Rightarrow A\vee(B\&C) \equiv (A\vee B)\&(A\vee C)$ 13, 23, пр. и

44. $\Rightarrow \neg(A\&B) \equiv \neg A\vee\neg B$

1. $A\&B \Rightarrow A\&B$ аксиома 2.
2 $A\&B \Rightarrow A$ 1, пр.2_1

3.¬A⇒¬A 1, аксиома
4. A&B,¬A⇒ 2, 3, пр.9
5.¬A,A&B ⇒ 4, пр. $12_{обобщ}$
6. ¬A⇒¬(A&B) 5 пр. 7
7. A&B⇒B 1, пр 2_2
8.¬B⇒¬B аксиома
9. A&B,¬B⇒ 7, 8, пр 9
10.¬B, A&B ⇒ 9, пр $10_{обобщ}$
11.¬B⇒¬(A&B) 10, пр 7
12.¬A∨¬B⇒¬(A&B) 6,11, пр г
13.¬A⇒¬A∨¬B 3, пр 3_1
14.¬B⇒¬A∨¬B 8, пр 3_2
15.¬(¬A∨¬B)⇒¬(¬A∨¬B) аксиома
16.¬A,¬(¬A∨¬B)⇒ 13, 15, пр.9
17.¬(¬A∨¬B),¬A⇒ 16, пр. $10_{обобщ}$
18.¬(¬A∨¬B)⇒A 17, пр. 8
19.¬B,¬(¬A∨¬B)⇒ 14, 15, пр. 9
20.¬(¬A∨¬B)⇒B 19, пр. 10, пр. 8
21.¬(¬A∨¬B)⇒A&B 18, 20, пр. 1
22.¬(A&B)⇒¬(A&B) аксиома
23.¬(¬A∨¬B),¬(A&B)⇒ 20, 22, пр. 9
24.¬(A&B),¬(¬A∨¬B)⇒ 23, пр. $10_{обобщ}$
25.¬(A&B)⇒¬A∨¬B 24, пр. 8
26.⇒¬(A&B)≡¬A∨¬B 12, 25, пр. и

45.⇒¬(A∨B)≡¬A&¬B аналогично 44

46.⇒A→B≡¬A∨B

1. A→B⇒A→B аксиома
2. A⇒A аксиома
3. A→B, A⇒B 1,2, пр. 6
4. A→B, A⇒¬A∨B 3, пр. 3_2
5. A→B,¬(¬A∨B)⇒¬A 4, пр. д
6. A→B,¬(¬A∨B)⇒¬A∨B 5, пр. 3_1
7. ¬(¬A∨B)⇒¬(¬A∨B) аксиома
8. A→B,¬(¬A∨B)⇒¬(¬A∨B) 7, пр. 11, пр. 10
9. A→B,¬(¬A∨B)⇒ 6, 8, пр. 9
10. A→B⇒¬A∨B 9, пр. 8
11. ¬A∨B⇒¬A∨B аксиома
12. ¬A⇒¬A аксиома
13. ¬A, A⇒ 12, 2, пр. 9
14. ¬A, A,¬B⇒ 13, пр. $11_{обобщ}$
15. ¬A,A ⇒B 14, пр. 8
16. ¬A⇒A→B 15, пр. 5
17. B⇒B аксиома
18. B,A⇒B 17, пр. 11
19. B⇒A→B 18, пр. 5
20. ¬A∨B⇒A→B 11, 16, 19, пр. 4
21. ⇒A→B≡¬A∨B 10, 20, пр. и

47.⇒¬A∨A

1. A→A⇒¬A∨A упр. 46.10, при B=A
2. A⇒A аксиома
3. ⇒A→A 2, пр. 5
4. ⇒¬A∨A 3, пр. а

48.⇒(A→B)∨(B→A)

1. B⇒B аксиома
2. B, ¬A, A⇒B 1, пр. 11, пр. 11
3.B, ¬A⇒A→B 2, пр. 5
4.¬A,B⇒A→B 3, пр. 10
5.¬A,¬(A→B)⇒¬B 4, пр. д
6.¬(A→B),¬A⇒¬B 5, пр. 10
7.¬(A→B),B⇒A 6, пр. е
8.¬(A→B)⇒B→A 7, пр. 5
9.¬(A→B),¬(B→A)⇒B→A 8, пр. 11
10.¬(A→B)&¬(B→A)⇒B→A 9, пр. ж
11.A→B⇒A→B аксиома
12.A→B⇒(A→B)∨(B∨A) 11, пр.3_1
13. ¬((A→B)∨(B∨A)) ⇒¬(A→B) 12, пр.д
14. B→A⇒B→A аксиома
15. B→A⇒(A→B)∨(B∨A) 14, пр.3_2
16.¬((A→B)∨(B∨A)) ⇒¬(B→A) 15, пр.д
17.¬((A→B)∨(B∨A)) ⇒¬(A→B)&¬(B→A) 13, 16, пр.1, пр.12
18. ¬((A→B)∨(B∨A)) ⇒B→A 17,10 пр.а
19. ¬((A→B)∨(B∨A)) ⇒ 18, 16, пр. 9
20. ⇒(A→B)∨(B→A) 19, пр.8

Теория алгоритмов

Алгоритм – первоначальное, интуитивно ясное понятие. Однако, для математического исследования этого понятия необходимо его формальное определение. В процессе развития теории алгоритмов некоторые подходы к определению алгоритма были отнесены к классическим. Чаше всего к таким подходам относят: машины Тьюринга, частично рекурсивные функции, канонические системы Поста, алгорифмы Маркова, РАМ(МНР)-машины, формально порождающие грамматики.

С методологической точки зрения мы будем использовать *тезис Черча* в следующей формулировке:

Все классические определения алгоритма эквивалентны между собой и эквивалентны интуитивному понятию алгоритма.

Тезис Черча позволяет сократить доказательства существования тех или иных формальных алгоритмов. Например, для того чтобы формально доказать эквивалентность определения алгоритма в терминах машины Тьюринга и в виде программ на языке Паскаль, мы обязаны предъявить два транслятора – один для перевода программы для машины Тьюринга в Паскаль-программу и транслятор Паскаль-программ в программы для машины Тьюринга. (Первая задача уровня студенческой на младших курсах, вторая – весьма трудоемкая задача, требующая, вообще говоря, знания методов синтаксического разбора и компиляции с языков программирования). Однако, существование подобных трансляторов не вызывает сомнения, поэтому доказательство эквивалентности этих двух определений алгоритма можно провести по тезису Черча. Естественно, если мы не можем представить интуитивно ясный алгоритм решения некоторой задачи, то применение тезиса Черча для доказательства его существования недопустимо.

Один из подходов к формальному определению алгоритма основан на представлении алгоритмов в виде программ для МНР-машины. Будем считать, областью исходных данных для алгоритмов являются подмножества в N^m, $m>0$.

Машина с непосредственным доступом к регистрам.

МНР-машина представляет собой предельно упрощенную абстрактную модель вычислительной машины (компьютера). Она имеет процессор и память. Процессор может выполнять только машинные команды четырех типов над значениями, хранящимися в памяти машины. Память представляет собой сколь угодно большое конечное множество регистров, в каждом из которых могут храниться натуральные числа. При этом каждый регистр определяется своим номером, а хранящееся в нем натуральное число может быть сколь угодно большим. Регистры занумерованы натуральными числами – 0,1, 2,… .

Любую конечную последовательность занумерованных команд будем называть программой для МНР-машины. Команды в программе, состоящей из k команд, занумерованы натуральными числами – 1, 2, …, k. Система команд МНР-машины представлена в следующей таблице

Тип команды	Команда	Ответ МНР
Обнуление	Z R	$0\to R$ (в регистр с номером R посылается значение 0)
Прибавление единицы	S R	$<R>+1\to R$ (значение, находящееся в регистре R, увеличивается на 1)
Пересылка значения	T R1 R2	$<R1>\to R2$ (значение, находящееся в регистре R1, записывается в регистр R2, при этом значение в регистре R1 также сохраняется)
Условный переход	J R1 R2 n	Если $<R1> = <R2>$, то выполняется команда программы, помеченная номером n, иначе – команда, следующая за командой условного перехода

Введем понятие “исполнение программы для МНР-машины”. Будем считать, что первой исполняется команда, помеченная номером 1, далее исполняются для команд первых трех типов команды, номер которых на единицу больше исполненной, а следующая команда после команды условного перехода зависит от условия, проверяемого в команде условного перехода. Исполнение программы заканчивается (останов программы) в случае попытки исполнить команду, номер которой не определен в программе. Перед исполнением МНР-программы ей необходимо задать исходные данные – некоторый вектор $x=(x_1,\ldots,x_m)\in N^m$. Для этого значения $x_1, \ldots, x_m$ помещают в первые m регистров с 0 по m-1 соответственно. Все остальные регистры, используемые в программе, в начальный момент считаются хранящими нулевые значения. Результатом исполнения МНР-программы считается значение, полученное в 0 регистре в случае останова МНР-программы.

Процесс исполнения МНР программ можно интерпретировать как процесс вычисления некоторой частичной функции $f : N^m\to N$

Введем обозначение. Пусть задана частичная функция $f:N^m\to N$, тогда $f(x)\uparrow$ обозначает, что функция f в точке x определена, а $f(x)\downarrow$ – не определена. Аналогично, если P некоторая МНР программа, то $P(x)\uparrow$ обозначает, что программа P с исходными данными x заканчивает работу, а $P(x)\downarrow$ не заканчивает работу, т. е. ‘зацикливается’.

Определение. Будем говорить, что частичная функция $f:N^m\to N$ вычислима, если существует МНР программа, ее вычисляющая. Вычислимые функции также называют алгоритмами, а если при этом они всюду определены, то полными алгоритмами.

Примеры.

1.Всюду неопределенная функция на N вычислима.

В самом деле, ее вычисляет следующая МНР программа 1. J 0 0 1

1. $f:N\to N, \forall x\in N\ f(x) = x \dot{-} 1 = \begin{cases} 0, & x = 0 \\ x-1, & x > 0 \end{cases}.$

Ее вычисляет следующая МНР программа

1. J 0 1 8
2. S 1
3. J 0 1 7
5. S 2

6. J 0 0 2
7 T 2 0

3 $f:N\to N$, $\forall x\in N$ $f(x)=k$, где $k\in N$

Ее вычисляет следующая МНР программа

$$\left.\begin{array}{l}1.\ Z\ 0\\ 2.\ S\ 0\\ \vdots\\ k+1.\ S\ 0\end{array}\right\} k \text{ раз}.$$

Упражнение. Докажите, что следующие функции вычислимы:

а) $f:N\to N$, $\forall x\in N$ $f(x)=x\dot{-}1=\begin{cases}\text{не определена, если } x=0\\ x\text{-}1, \text{ если } x>0\end{cases}$.

б) $f:N\to N$, $\forall x\in N$ $f(x)=2x$.

в) $f:N^2\to N$, $\forall x,y\in N$ $f(x,y)=x+y$.

г) $f:N^2\to N$, $\forall x,y\in N$ $f(x,y)=x\dot{-}y=\begin{cases}0, \text{ если } x<y\\ x\text{-}y, \text{ если } x\ge y\end{cases}$.

Свойства МНР программ.

1. Множество вычислимых функций не более чем счетно.

Каждая программа конечный текст в конечном алфавите.

Следствие. Существуют невычислимые функции.

Для доказательства заметим, что существует континиумм функций вида $f:N\to\{0,1\}$.

2. Существует счетное множество программ, вычисляющих конкретную вычислимую функцию.

Для доказательства заметим, что если в конец любой программы приписать команду вида T k k, где k номер любого регистра, то так преобразованная программа вычисляет ту же функцию, что до преобразования.

Программы будем называть эквивалентными, если они вычисляют одну и ту же функцию.

3. Для дальнейших исследований свойств программ нам будет полезно преобразование МНР-программы в стандартную форму.

Определение. МНР-программу из m команд будем считать находящейся в стандартной форме, если в каждой ее команде перехода с третьим операндом, большим m, этот операнд имеет значение m+1. Кроме того, считаем, что при окончании работы программы в стандартной форме, все регистры памяти за исключением нулевого обнуляются.

Очевидно , что для любой программы существует ей эквивалентная, представленная в стандартной форме. В дальнейшем будем считать, что рассматриваемые программы представлены в стандартной форме.

Кроме того, в дальнейшем для нас будет представлять интерес преобразование текста МНР-программы, связанное со смещением ее начального адреса. Пусть P некоторая МНР-программа, состоящая из m команд. Увеличим на $l\in N$ номера всех команд программы P, а также третий операнд во всех командах перехода программы P. Так преобразованную программу P будем обозначать P[l] и называть программой P со смещенным начальным адресом на величину l. Отметим, что результат исполнения последовательности команд P[l] совпадает с результатом исполнения программы P, при условии, что первой для P[l] исполняется команда с номером l+1.

При конструировании новых программ из фрагментов других программ мы часто будем писать конструкции вида P[*]. В этом случае значение, соответствующее *, совпадает с номером предыдущей команды конструируемой программы.

Следующую последовательность команд будем обозначать COPY(a,b,m).

1 T a+0 b+0
. . .
j T a+j b+j $\quad 0 \le j < m$
. . .
m-1 T a+m-1 b+m-1

4. Пусть $F:N^k \to N$ и $f_1,..,f_k:N^n \to N$ вычислимые функции, тогда их <u>суперпозиция</u> – функция $h:N^n \to N$ такая, что $\forall x=(x_1,\dots x_n) \in N^n$ $h(x)=F(f_1(x),\dots,f_k(x))$, также вычислима. Обозначение $h=S(F,f_1,..,f_k)$.

Доказательство.

Пусть вычислимые функции F, $f_1,\dots, f_k$ порождаются программами P_0, P_1, …, P_k соответственно. Каждая из этих программ соответственно использует в процессе вычисления по ней первые m_0, m_1, …, m_k регистры. Пусть $m>\max(m_0, m_1, \dots, m_k)$, тогда следующая программа вычисляет функцию h:

1.COPY(0,m,n)
. P_1[*]
. T 0 m+n
. COPY(m,0,n)[*]
. P_2[*]
. T 0 m+n+1
.
.
.
. COPY(m,0,k)[*]
. P_k[*]
. T 0 m+n+k-1
. COPY(m+n,0,k)[*]
. P_0[*]

5. <u>Определение</u>. Пусть $f:N^n \to N$, $g:N^{n+2} \to N$, тогда говорят, что функция $h:N^{n+1} \to N$ получена из f и g с помощью <u>примитивной рекурсии</u>, если $\forall x=(x_1,\dots x_n) \in N^n$ и $y \in N$

$$h(x,0)=f(x)$$
$$h(x,y+1)=g(x,y,h(x,y)).$$

Обозначение $h=PR(f,g)$.

<u>Теорема</u>. Пусть $f:N^n \to N$, $g:N^{n+2} \to N$, $h:N^{n+1} \to N$, $h=PR(f,g)$, если f,g вычислимы, то и h – вычислима.

Доказательство.

Пусть вычислимые функции f, g порождаются программами P_0 и P_1, соответственно. Программа P_0 в процессе вычисления по ней использует первые m_0 регистров, а P_1 – m_1 регистров. Пусть $m>\max(m_0, m_1)$, тогда следующая программа вычисляет функцию h:

```
          1.COPY(0,m,к+1)
          . Z k
          . P0[*]
     цикл  T 0 k+1
          . J k m+k конец
          . S k
          . COPY(m,0,к)[*]
          . P1[*]
          . J 0 0 цикл
    конец
```

<u>Комментарий</u>. Метки 'цикл' и 'конец' определяются номерами соответствующих команд.

<u>Замечание</u>. Пусть $h_1=S(F,f_1,..,f_k)$ и $h_2=PR(f,g)$, если $F,f_1,..,f_k,f,g$ всюду определены, h_1 и h_2 также всюду определены.

6. <u>Определение</u>. Пусть $f:N^{n+1} \to N$, : тогда говорят, что функция $h:N^n \to N$ получена из f с помощью оператора <u>минимизации</u>, если $\forall x=(x_1,\dots x_n) \in N^n$

$$h(x)=\mu_y(f(x,y))=\begin{cases} \text{наименьшее } z\in N:(f(x,z)=0)\&(\forall u<z \Rightarrow f(x,u)\uparrow) \\ \text{не определено, если такого } z \text{ нет} \end{cases}.$$

<u>Теорема</u>. Пусть $f:N^{n+1} \to N$, f вычислима, тогда функция h, полученная из f с помощью оператора минимизации, также вычислима.

<u>Упражнение</u>. Постройте программу, вычисляющую функцию h.

<u>Замечание</u>. Оператор минимизации не сохраняет свойство всюду определенности функций.

<u>Пример</u>. Функция $f:N^2 \to N$, $\forall x,y\in N$ $f(x,y) = x+y$ всюду определена, но $\mu_y(f(x,y))$ определена только при x=0.

Примитивно рекурсивные функции.

Класс <u>примитивно рекурсивных функций</u> будем обозначать *PRC*.

<u>Определение</u>. $h\in$ *PRC*, если

1. h=zero $\{zero(x)= 0 \ \ \forall x\in N\}$
2. h=succ $\{ succ(x)= x+1 \ \ \forall x\in N\}$
3. $h=u_i^n$ $\{u_i^n:N^n \to N, \forall (x_1,\dots,x_n)\in N^n \ u_i^n(x_1,\dots,x_n)=x_i\}$
4. $h=S(f,g_1,\dots,g_k)$, где $f,g_1,\dots,g_k\in$ *PRC*
5. $h=PR(f,g)$ где $f,g\in$ *PRC*
6. других функций в классе *PRC* нет.

Функции zero, succ, u_i^n будем называть <u>первоначальными</u> или <u>простейшими</u>.

<u>Примеры</u>.

1.$h:N^2 \to N$, $h(x,y)=+(x,y)=x+y$. Покажем, что $h\in$ *PRC*.

Воспользуемся примитивной рекурсией

$h(x,0)=x+0=x \quad \Rightarrow \quad f(x)=x \quad \Rightarrow \quad f=u_1^1$

$h(x,y+1)=x+(y+1)=(x+y)+1 \ \Rightarrow\ g(x,y,z)=z+1 \ \Rightarrow\ g=\text{succ}(u_3^3)$,

таким образом, $h=PR(u_1^1, S(\text{succ}, u_3^3))$

2.$h:N^2 \to N$, $h(x,y)=\cdot(x,y)=x\cdot y$. Покажем, что $h\in$ *PRC*.

Воспользуемся примитивной рекурсией

$h(x,0)=x\cdot 0=0 \quad \Rightarrow \quad f(x)=0 \quad \Rightarrow \quad f=\text{zero}$

$h(x,y+1)=x\cdot(y+1)=(x\cdot y)+y \ \Rightarrow\ g(x,y,z)=z+y \ \Rightarrow\ g=+(z,y)$,

таким образом, $h=PR(\text{zero}, +(S(u_3^3), S(u_2^3)))$

3.$h:N^2 \to N$, $h(x,y)=x^y$. Покажем, что $h\in$ *PRC*.

Воспользуемся примитивной рекурсией

$h(x,0)=x^0=1 \quad \Rightarrow \quad f(x)=1 \quad \Rightarrow \quad f=\text{succ}(\text{zero})$

$h(x,y+1)=x^{(y+1)}=(x^y)\cdot x \ \Rightarrow\ g(x,y,z)=z\cdot x \quad \Rightarrow \quad g=\cdot\,(z,x)$,

таким образом, $h=PR(\text{succ}(\text{zero}), \cdot(S(u_3^3), S(u_1^3)))$

4. $h:N \to N$, $h(x)=k$, $k\in N$ Покажем, что $h\in$ *PRC*.

$h=\underbrace{\text{succ}(\text{succ}(\ldots(\text{succ}}_{k\ \text{раз}}(\text{zero}))\ldots))$

5. $h:N \to N$, $h(x)=\dot-(x,1)=x\dot-1=\begin{cases}0, \text{если } x=0\\ x-1, \ \text{если } x>0\end{cases}$. Покажем, что $h\in$ *PRC*.

Воспользуемся примитивной рекурсией по x, считая n=0 в определении примитивной рекурсии.

$0\dot-1=0 \quad \Rightarrow \quad f(\,)=0$

$(x+1)\dot-1=x \quad \Rightarrow \quad g(x,y)=x \ \Rightarrow\ g=u_1^2$

6. $h:N^2 \to N$, $h(x)=x\dot-y=\begin{cases}0, \text{если } x<y\\ x-y, \text{если } x\geq y\end{cases}$. Покажем, что $h\in$ *PRC*.

Воспользуемся примитивной рекурсией по y,

$x\dot-0=x \quad \Rightarrow \quad f(x)=x \quad \Rightarrow \quad f=u_1^1$

$(x)\dot-(y+1)=(x\dot-y)\dot-1 \quad \Rightarrow \quad g(x,y,z)=z\dot-1 \ \Rightarrow\ g=\dot-(u_3^3,1)$

7. $h:N \to N$, $h(x)=sg(x)=\begin{cases}0, \text{если } x=0\\ 1, \ \text{если } x\neq 0\end{cases}$. Покажем, что $h\in$ *PRC*.

Воспользуемся примитивной рекурсией по x, считая n=0 в определении примитивной рекурсии.

$sg(0)=0 \quad \Rightarrow \quad f(\,)=0$

$sg(x+1)=1 \quad \Rightarrow \quad g(x,y)=1 \ \Rightarrow\ g=\text{succ}(\text{zero}(u_1^2))$

8. $h:N \to N$, $h(x)=\overline{sg}(x)=\begin{cases}1, \text{если } x=0\\ 0, \ \text{если } x\neq 0\end{cases}$. Покажем, что $h\in$ *PRC*.

Кроме решения, аналогичного решению примера 7, можно предложить решение $\overline{sg}(x)=1\dot-sg(x)$.

<u>Упражнение</u>. Запишите $\overline{sg}$ в операторной форме.

Функция $\overline{sg}$ обладает свойством $\forall x(\overline{sg}(x) \neq x)$

9. $h: N^2 \to N$, $h(x,y)=|x-y|$

$h \in$ *PRC*, так как $|x-y|=(x \dot{-} y)+(y \dot{-} x)$

10. $h: N \to N$, $h(x)=x!$

Принадлежность h к классу *PRC* доказывается примитивной рекурсией по x.

11. $h: N^2 \to N$, $h(x,y)=\min\{x-y\}$

$h \in$ *PRC*, так как $\min\{x,y\}=x \dot{-} (x \dot{-} y)$

12. $h: N^2 \to N$, $h(x,y)=\max\{x-y\}$

$h \in$ *PRC*, так как $\max\{x-y\}=x+(y \dot{-} x)$

13. $h: N^2 \to N$, $h(x,y)=y \bmod x$ {если x=0, то $y \bmod 0 = y$}

Принадлежность h к классу *PRC* докажем примитивной рекурсией по y.

Имеем $(y+1) \bmod x = \begin{cases} y \bmod x + 1, \text{если } y \bmod x + 1 \neq x \\ 0, \quad \text{если } y \bmod x + 1 = x \end{cases}$.

Это приводит

$0 \bmod x = 0$

$(y+1) \bmod x = (y \bmod x + 1) \cdot sg(|x-(y \bmod x + 1)|)$

т. е. $f(x)=0$

$g(x,y,z)=(z+1) \cdot sg(|x-(z+1)|)$.

14. $h: N^2 \to N$, $h(x.y)=y \operatorname{div} x$ {если x=0, то $y \operatorname{div} 0 = 0$}

Принадлежность h к классу *PRC* докажем аналогично 13.

Имеем $(y+1) \operatorname{div} x = \begin{cases} y \operatorname{div} x, \text{если } y \bmod x + 1 \neq x \\ y \operatorname{div} x + 1, \quad \text{если } y \bmod x + 1 = x \end{cases}$.

Это приводит

$0 \operatorname{div} x = 0$

$(y+1) \operatorname{div} x = (y \operatorname{div} x) + \overline{sg}(|x-(y \bmod x + 1)|)$

т. е. $f(x)=0$

$g(x,y,z)=z+\overline{sg}(|x-(y \bmod x+1)|)$.

15. $d: N^2 \to N$, $d(x,y)=\begin{cases} 1, \text{если } x \mid y \ \{x \text{ делит } y\}, & \text{считаем } 0 \mid 0, \\ 0, \quad \text{если } x \nmid y \ \{x \text{ не делит } y\} & 0 \nmid y, \text{ если } y \neq 0 \end{cases}$

очевидно, что $d(x,y)=\overline{sg}(y \bmod x)$, т.е. $d \in$ *PRC*.

16. $D: N \to N$, $D(y)=\sum_{x \leq y} d(x,y)$, считаем, что $D(0)=1$.

Доказательство принадлежности D к классу *PRC* основано на следующей теореме:

<u>Теорема 1</u>. Пусть $x=(x_1,\ldots,x_n) \in N^n$, $y,z \in N$, $f: N^{n+1} \to N \in$ *PRC*, тогда

$\sum_{z<y} f(x,z) \in PRC$ и $\prod_{z<y} f(x,z) \in PRC$.

Доказательство проведем рекурсией по y.

$\sum_{z<y} f(x,z) = 0$ $\qquad$ $\prod_{z<0} f(x,z) = 1$

$\sum_{z<y+1} f(x,z) = (\sum_{z<y} f(x,z)) + f(x,y)$ $\qquad$ $\prod_{z<y+1} f(x,z) = (\prod_{z<y} f(x,z)) \cdot f(x,y))$

Следствие. Если f(x,z) и k(x,w) $\in PRC$, w$\in$N, то $\sum_{z<k(x,w)} f(x,z)$ и $\prod_{z<k(x,w)} f(x,z) \in PRC$.

17. Pr : N→N, $Pr(x)=\begin{cases} 1, \text{ если x-простое} \\ 0, \text{ если x- не простое} \end{cases}$.

Заметим, что Pr (x)= $\overline{sg}$ (|D(x)-2|), т.е. Pr$\in PRC$.

Для доказательства примитивной рекурсивности некоторых функций весьма полезен оператор ограниченной минимизации.

Определение. Пусть, $f:N^{n+1} \to N$ тогда говорят, что функция $g:N^{n+1} \to N$, такая что $\forall$ $x=(x_1,\ldots,x_n) \in N^n$ и $\forall$ $y \in N$

$$g(x,y)=\begin{cases} \text{наименьшее } z: z < y,\ f(x,z) = 0, \text{ если такое z существует} \\ y, \text{ если такого z нет} \end{cases}$$

получена из f с помощью оператора ограниченной минимизации. Обозначение $g(x,y)=\mu_{z<y} f(x,z)$.

Теорема 2. Пусть, $f:N^{n+1} \to N$ и $f \in PRC$, тогда $g(x,y)= \overline{\mu_{z<y}} f(x,z) \in PRC$.

Доказательство.

$$\overline{\mu_{z<y}} f(x,z) = \sum_{u<y} (\prod_{z \le u} sg(f(x,z))).$$

Замечание. Наряду с оператором ограниченной минимизации для функций в теории алгоритмов рассматриваются операторы ограниченной минимизации для предикатов.

Определение. Пусть Q предикат на N^n, тогда функцию $q:N^n \to N$ называют представляющей функцией предиката Q, если

$\forall$ $x=\in N^n$ $q(x)=\begin{cases} 0, \text{ если } Q(x) \\ 1, \text{ если } \neg Q(x) \end{cases}$. $\qquad \overline{\mu_{z<y}} Q(x,z) \overset{def}{=} \overline{\mu_{z<y}} q(x,z)$.

18. h: N→N, $h(x)=\lfloor \sqrt{x} \rfloor$[1]

h(x)$\in PRC$, так как $h(x)= \overline{\mu_{z<x}}$ $(sg((z+1)^2-x)=1)$

19.P : N→N, P(x)=x-простое число {P(0)=0}

Принадлежность h к классу PRC докажем примитивной рекурсией по x

P(0)=0

[1] $\lfloor x \rfloor$ обозначает целую часть вещественного числа x.

$P(x+1)=\overline{\mu_{z\le P(x)!+1}}((z>P(x))\&(z$ – простое число)).

20.h:$N^2\to N$,

$h(x,y)=exp_y x$ – показатель $P(y)$ в разложении x на простые множители.

$h(x,y)\in$ *PRC,* так как $h(x,y)=\overline{\mu_{z<y}}\,(P^{z+1}(y)\dagger x)$.

21. F : $N\to N$, F(0)=1, F(1)=2 и $\forall n>0$ F(n+2)=F(n+1)+F(n).

Принадлежность F к классу *PRC* следует из леммы:

<u>Лемма</u>. $\forall n>0\;\; F(n)=\sum_{k=0}^{\lfloor (n+1)/2\rfloor} C_k^{n-k+1}=\sum_{k=0}^{\lfloor (n+1)/2\rfloor}\frac{\lfloor (n+1)/2\rfloor!}{k!(\lfloor (n+1)/2\rfloor-k)!}$

Доказательство леммы проведем методом комбинаторного (теоретико-множественного) доказательства. Для этого рассмотрим множество последовательностей из нулей и единиц длины n, в которых нет двух рядом стоящих единиц. По индукции покажем, что для заданного n число таких последовательностей есть F(n). Пусть число таких последовательностей длины n есть A(n).

Тогда, A(0)=1, так как существует только одна пустая такая последовательность; A(1)=2, так как существуют две такие последовательности – ‘0’ и ‘1’.

Индукционное предположение для $k\le n$ A(n)=F(n).

Заметим, что число последовательностей длины n, у которых на n месте находится нуль, равно A(n-1), т. е. F(n-1).

Все последовательности длины n+1 могут быть построены из последовательностей длины n приписыванием к каждой из них на n+1 место нуля и, кроме того, тем из них, которые на n месте имеют ноль, можно также приписать единицу. Таким образом, A(n+1)=A(n)+A(n-1)=F(n)+F(n-1)=F(n+1).

С другой стороны, A(n) можно получить следующим образом:

Заметим, каждая такая последовательность длины n может содержать не более $\lfloor (n+1)/2\rfloor$ единиц. Подсчитаем, сколько существует последовательностей, содержащих k единиц, $0\le k\le\lfloor (n+1)/2\rfloor$. Если последовательность имеет к единиц, то она содержит n-k нулей. Рассмотрим последовательность из n-k нулей. Тогда в этой последовательности имеется n-k+1 мест для расстановки к единиц. Т. е. общее число требуемых последовательностей длины n,содержащих k единиц, равно C_k^{n-k+1}. Таким образом, $A(n)=\sum_{k=0}^{\lfloor (n+1)/2\rfloor} C_k^{n-k+1}$.

<u>Замечание</u>. Решение задачи 21 может быть получено на основе теоремы 3(о возвратной рекурсии).

<u>Возвратная рекурсия</u>.

<u>Определение</u>. Пусть $x=(x_1,\ldots,x_n)\in N^n$, $y,z_1,\ldots,z_s\in N$, $f:N^n\to N$, $g:N^{n+s+1}\to N$, $\alpha_1,\ldots,\alpha_s:N\to N$, при этом $\forall y\in N$ $\alpha_i(y)<y$, $1\le i\le s$, тогда функцию $h:N^{n+1}\to N$, такую что

h(x,0)=f(x)

$h(x,y+1)=g(x,y,\ h(x,\alpha_1(y+1)),\ldots,\ h(x,\alpha_s(y+1)))$

называют функцией, полученной из $f,g,\alpha_1,\ldots,\alpha_s$ с помощью <u>возвратной рекурсии</u>.

<u>Теорема</u> 3. Если $f,g,\alpha_1,\ldots,\alpha_s\in PRC$, то $h\in PRC$.

Доказательство.

Рассмотрим функцию $H(x,y)=\prod_{i=0}^{y} P^{h(x,i)}(i+1)$, тогда

$$\forall i\le y \quad h(x,i) = \exp_{i+1}(H(x,y)). \qquad \{*\}$$

По условию $\alpha_j(y+1)\le y$, поэтому $h(x,\alpha_j(y+1))=\exp_{\alpha_j(y+1)+1}H(x,y)$.

Имеем $H(x,0)=P^{f(x)}(1)$ $H(x,y+1)=H(x,y)\cdot P^{h(x,y+1)}((y+1)+1)$

т. е. $H(x,y+1)= H(x,y)\cdot P^{g(x,y,h(x,\alpha_1(y+1)),\ldots,h(x,\alpha_s(y+1)))}((y+1)+1)$.

Пусть $F(x)= P^{f(x)}(1)$,

$$G(x,y,z)=z\cdot P^{g(x,y,\exp_{\alpha_1((y+1)+1)}z,\ldots,\exp_{\alpha_s((y+1)+1)}z)}((y+1)+1),$$

тогда $F,G\in PRC$, кроме того

$H(x,0)=F(x)$

$H(x,y+1)=G(x,y,H(x,y))$

т. е. $H\in PRC$.

Учитывая *, получаем $h(x,y)=\exp_{y+1}H(x,y)$, т. е. $h\in PRC$.

При решении задач на доказательство примитивной рекурсивности конкретных функций может быть полезна следующая теорема:

<u>Теорема 4</u>. Пусть $f_1,\ldots,f_s,f_{s+1},\alpha_1,\ldots,\alpha_s:N^n\to N \in PRC$, при этом ни при каких значениях аргумента никакие две функции $\alpha_1,\ldots,\alpha_s$ не обращаются в нуль одновременно. Тогда функция $f:N^n\to N$, определенная схемой

$$\forall x\in N^n\ f(x)=\begin{cases} f_1(x), \text{ если } \alpha_1(x)=0,\\ \ldots\\ f_s(x), \text{ если } \alpha_s(x)=0,\\ f_{s+1}(x), \text{ в остальных случаях.}\end{cases}$$

будет также примитивно рекурсивной.

Для доказательства достаточно заметить, что функцию f можно представить в виде $f=f_1\overline{sg}(\alpha_1)+\ldots+f_s\overline{sg}(\alpha_s)+f_{s+1}sg(\alpha_1\ldots\alpha_s)$

22. {последовательность Фибоначчи} $\Phi : N\to N$, $\Phi(0)=0$, $\Phi(1)=1$ и $\forall n>0$ $\Phi(n+2)=\Phi(n+1)+\Phi(n)$.

Принадлежность Φ к классу *PRC* следует из решения задачи 21 и теоремы 4. Кроме того, решение задачи следует непосредственно из теоремы о возвратной рекурсии.

23. Примитивно рекурсивные функции могут быть представлены весьма экзотично:

$$C : N\to N,\ C(n)= \frac{1}{2\pi}\int_0^{2\pi}\frac{\sin^{2n}10x}{\sin^{2n}x}dx \ \Rightarrow\ C\in PRC$$

Доказательство легко получить из леммы

Лемма . Докажем, что число счастливых 2n-значных трамвайных билетов равно

$$\frac{1}{2\pi}\int_0^{2\pi}\frac{\sin^{2n}10x}{\sin^{2n}x}dx .$$

Билет считается счастливым, если сумма первых n цифр его номера равна сумме n последних цифр, например, билет с номером 764395 – счастливый шестизначный билет.

Упражнение. Докажите, что функция от параметра n, вычисляющая число счастливых 2n-значных трамвайных билетов, примитивно рекурсивна.

Доказательство (леммы).

Рассмотрим равенство $(1+z+\ldots+z^9)^n=\sum_{i=0}^{9n} a_i z^i$, тогда a_i определяет количество n-значных чисел, сумма цифр которых равна i.

Нам нужно вычислить $\sum_{i=0}^{9n} a_i^2$.

Имеем $(1+z+\ldots+z^9)^n(1+z^{-1}+\ldots+z^{-9})^n=\sum_{i=0}^{9n} a_i z^i \times \sum_{i=0}^{9n} a_i z^{-i} = \sum_{m=-9n}^{9n} b_m z^m$,

тогда $b_0=\sum_{i=0}^{9n} a_i^2$.

$$(1+z+\ldots+z^9)^n(1+z^{-1}+\ldots+z^{-9})^n=\left(\frac{1-z^{10}}{1-z}\cdot\frac{1-z^{-10}}{1-z^{-1}}\right)^n=\frac{(2-z^{10}-z^{-10})^n}{(2-z-z^{-1})^n} .$$

Известно, что $\frac{1}{2\pi}\int_0^{2\pi} e^{im\phi}d\phi=\begin{cases}1, \text{если } m=0\\ 0, \text{если } m\neq 0\end{cases}$.

Пусть $z=e^{i\varphi}=\cos\varphi+i\sin\varphi$, тогда $b_0=\sum_{i=0}^{9n} a_i^2=\frac{1}{2\pi}\int_0^{2\pi}\sum_{m=-9n}^{9n} b_m e^{im\phi}d\phi=\frac{1}{2\pi}\int_0^{2\pi}\frac{(2-e^{i10\phi}-e^{-i10\phi})^n}{(2-e^{i\phi}-z^{-i\phi})^n}d\phi$.

Преобразуем $\frac{(2-e^{i10\phi}-e^{-i10\phi})}{(2-e^{i\phi}-z^{-i\phi})}=$

$$\frac{2-\cos 10\phi-i\sin 10\phi-\cos(-10\phi)-i\sin(-10\phi)}{2-\cos\phi-i\sin\phi-\cos(-\phi)-i\sin(-\phi)}=$$

$$\frac{2-2\cos 10\phi}{2-2\cos\phi}=\frac{1-\cos 10\phi}{1-\cos\phi}=\frac{\sin^2(10\phi/2)}{\sin^2(\phi/2)} .$$

Таким образом,

$$b_0=\frac{1}{2\pi}\int_0^{2\pi}\frac{\sin^{2n}(10\phi/2)}{\sin^{2n}(\phi/2)}d\phi=\frac{1}{\pi}\int_0^{\pi}\frac{\sin^{2n}10x}{\sin^{2n}x}dx=\frac{1}{2\pi}\int_0^{2\pi}\frac{\sin^{2n}10x}{\sin^{2n}x}dx.$$

Частично рекурсивные функции.

Класс частично рекурсивных функций будем обозначать *RC*.

Определение. Функция h принадлежит классу *RC*, если она совпадает с одной из первоначальных или получена из частично рекурсивных с помощью операторов суперпозиции, примитивной рекурсии и минимизации.

Справедлива следующая теорема:

Класс вычислимых функций совпадает с классом *RC*.

Определение. Всюду определенные функции из *RC* называют общерекурсивными.

Можно доказать, что класс общерекурсивных функций шире класса примитивно рекурсивных функций. Доказательство этой теоремы может быть построено на изучении понятия B-мажорируемости.

Определение.(B-функции). Пусть $B:N^2\to N$ и $\forall x,y\in N$ справедливо:

$$B(0,y)=2+y$$
$$B(x+1,0)=sg(x)$$
$$B(x+1,y+1)=B(x,B(x+1,y)).$$

Функцию $A:N\to N$ такую, что $\forall x\in N$ $A(x)=B(x,x)$, называют функцией Аккермана.

Определение. Всюду определенную функцию $f : N^k\to N$ называют B-мажорируемой, если существует $m\in N$ такое, что

$$\forall x=(x_1,\ldots,x_k)\in N^k \quad f(x_1,\ldots,x_k)<B(m, \max(x_1,\ldots,x_k)).$$

Упражнение. 1. Докажите, что функции B и A общерекурсивны.

2.$\forall$ x,y$\in$N а)$B(y+2,x+1)\geq 2^{x+1}$;

б)$B(y+1,x+2)\geq B(y+1,y+1)$;

в)$B(y+2,x+2)\geq B(y+1,y+3)$.

3. Первоначальные функции B-мажорируемы.

4. Функция, полученная с помощью оператора суперпозиции из B-мажорируемых функций, B-мажорируема.

5. Функция, полученная с помощью оператора примитивной рекурсии из B-мажорируемых функций, B-мажорируема.

6. Функция Аккермана не является примитивно рекурсивной.

Примитивно рекурсивные соответствия.

Определение. Пусть задано биективное соответствие $N\sim N^2$ такое, что натуральному n сопоставляется пара (x,y). Пусть функции $c:N^2\to N$, $l,r:N\to N$ такие, что $c(x,y)=n$, $l(n)=x$ и $r(n)=y$. Это соответствие называют примитивно рекурсивным, если примитивно рекурсивны функции c,l,r.

Замечание. В теории алгоритмов подобные биективные соответствия часто называют нумерацией множества N^2.

Докажем существование примитивно рекурсивных соответствий $N \sim N^2$.

а.(соответствие Кантора).

y\x	0	1	2	3	4	5	. . .	x	. . .
0	0	2	5	9	14	20		⋮	
1	1	4	8	13	19				
2	3	7	12	18				⋮	
3	6	11	17						
4	10	16							
5	15							⋮	
⋮									
y	. . .	. . .	. . .	. . .	. . .	. . .	. . .	n	
⋮									

<u>Упражнение</u>. Докажите, что при соответствии Кантора справедливо: $n=c(x,y)=(x+y)(x+y+1)/2+x$.

Покажем, что $x=l(n)=n \mathbin{\dot-} \frac{1}{2}\cdot\left[\frac{\lfloor\sqrt{8n+1}\rfloor+1}{2}\right]\cdot\left[\frac{\lfloor\sqrt{8n+1}\rfloor \mathbin{\dot-} 1}{2}\right]$

Имеем, $$2n=(x+y)^2+3x+y$$

или $$8n+1=(2x+2y+1)^2+8x=(2x+2y+3)^2-8y-8,$$

тогда $$2x+2y+1\le\lfloor\sqrt{8n+1}\rfloor<2x+2y+3$$

$$x+y+1\le\frac{\lfloor\sqrt{8n+1}\rfloor+1}{2}< x+y+2,$$

т. е. $x+y+1=\left[\frac{\lfloor\sqrt{8n+1}\rfloor+1}{2}\right]$, отсюда $x+y=\left[\frac{\lfloor\sqrt{8n+1}\rfloor \mathbin{\dot-} 1}{2}\right]$

и $$x=n \mathbin{\dot-} \frac{1}{2}\cdot\left[\frac{\lfloor\sqrt{8n+1}\rfloor+1}{2}\right]\cdot\left[\frac{\lfloor\sqrt{8n+1}\rfloor \mathbin{\dot-} 1}{2}\right].$$

<u>Упражнение</u>. Выразите y как функцию от n.

б.(соответствие Геделя).

y\x	0	1	2	3	. . .	x	. . .
0	0	1	3	7			
1	2	5	11	23		⋮	
2	4	9	19	39			
3	6	13	27	55		⋮	
4	8	17	35	71			
⋮	⋮	⋮	⋮	⋮		⋮	

y	2y	4y+1	8y+3	16y+7	. . .	n	. . .
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$		$\vdots$	

<u>Упражнение</u>. Докажите, что при соответствии Геделя справедливо:

$$n=c(x,y)=2^x(2y+1)-1$$

$$x=l(n)=\exp_1(n+1)$$

$$y=r(n)=\frac{1}{2}((n+1)/(2^{\exp_1(n+1)}-1)).$$

Определив примитивно рекурсивную нумерацию N^2, легко построить биективное соответствие $N\sim N^k$ для $k>2$, например:

Нумерацию N^k обозначим c_k и определим эту функцию рекурсией по k:

$$n=c_k(x_1,x_2,\ldots,x_k)=c(x_1,c_{k-1}(x_2,\ldots,x_k));$$

причем для единообразия считаем, что $c_2=c$ и $c_1=u_1^1$. Обратные функции определяются очевидным образом:

$$l_k^1(n)=l(n)\quad l_k^2(n)=l(r(n))\ \ldots\ l_k^{k-1}(n)=l(\underbrace{r(\ldots r(n)\ldots)}_{k-1\text{ раз}})\quad l_k^k(n)=\underbrace{r(r(\ldots r(n)\ldots))}_{k\text{ раз}}$$

<u>Определение</u>. (Нумерация $\bigcup_{k>0} N^k$) Пусть задано биективное соответствие $N\sim\bigcup_{k>0} N^k$ такое, что натуральному n сопоставляется вектор $(x_1,x_2,\ldots,x_k)$. Пусть $\Phi:N^k\to N$ и $\phi_0:N\to N$ такие, что $\Phi(x_1,x_2,\ldots,x_k)=n$, $\phi_0(n)=k$, кроме того, для заданного k определяются функции $\varphi_1^k,\varphi_2^k,\ldots,\varphi_k^k: N\to N$ такие, что $\varphi_i^k(n)=x_i$, $1\le i\le k$. Это соответствие называют примитивно рекурсивным, если примитивно рекурсивны функции $\Phi,\phi_0,\varphi_1^k,\varphi_2^k,\ldots,\varphi_k^k$.

<u>Упражнение</u>. Докажите, что примитивно рекурсивным является биективное соответствие $N\sim\bigcup_{k>0} N^k$, которое вектору $(x_1,x_2,\ldots,x_k)$ сопоставляет значение $2^{x_1}+2^{x_1+x_2+1}+\ldots+2^{x_1+x_2+\ldots x_k+k-1}-1$.

Нумерация (геделизация) программ и вычислимых функций.

Зафиксируем произвольные примитивные рекурсивные соответствия:

$N\sim N^2$	$N\sim N^k$	$N\sim\bigcup_{k>0} N^k$
$\mathbf{c}: N^2\to N$	$\mathbf{c}_k: N^k\to N$	$\Phi:\bigcup_{k>0} N^k\to N$
$l,r:N\to N$	$l_k^i:N\to N$, $1\le i\le k$	$\varphi_0,\varphi_1^k,\ldots,\varphi_k^k: N\to N$
$\mathbf{c}(x,y)=n$	$\mathbf{c}_k(x_1,\ldots,x_k)=n$	$\Phi(x_1,\ldots,x_k)=n$
$l(n)=x$	$l_k^i(n)=x_i$	$\phi_0(n)=k$
$r(n)=y$		$\varphi_i^k(n)=x_i$, $1\le i\le k$

Сначала построим биекцию между командами МНР-машины и N:

Команде вида Z R сопоставим число 4R;

Команде вида S R сопоставим число 4R+1;

Команде вида T R1 R2 сопоставим число 4c(R1,R2)+2;

Команде вида J R1 R2 n сопоставим число $4c_3(R1,R2,n)+3$.
С помощью этого соответствия каждой МНР-программе P, содержащей k команд, можно сопоставить взаимно однозначно последовательность натуральных чисел $x_1,\ldots,x_k$, где x_i, $1\le i\le k$ определяется видом i-той команды программы P. Далее, вектору $(x_1,\ldots,x_k)$ сопоставим значение $e=\Phi(x_1,\ldots,x_k)$. Так построенное по программе P натуральное число e будем называть <u>геделевым номером</u> (или просто номером) <u>программы P</u>.

Заметим, что по любому номеру e легко восстановить программу P, ему соответствующую, и обратно. Если $e\in N$, тогда программу P, номер которой e, будем обозначать

$$e=\nu P \text{ или } P=\overline{\nu}e.$$

По отношению к некоторой вычислимой функции f в множестве N можно выделить его счетное подмножество номеров программ, вычисляющих функцию f. Под геделевым номером вычислимой функции f будем понимать любой из номеров программ, вычисляющих f. Также допустимыми будут обозначения:

$$e=\nu f \text{ или } f=\overline{\nu}e.$$

Как уже отмечалось, МНР-программы P и Q эквивалентны, обозначение $P\approx Q$, если они вычисляют одну и ту же функцию. Вполне допустимы обозначения: $e\approx\nu P$, $P\approx\overline{\nu}e$ или $e\approx\nu f$, $f\approx\overline{\nu}e$.

Теорема Клини о нормальной форме.

$t_f^*(x)$ – временная вычислительная сложность программы P над аргументом $x=(x_1,x_2,\ldots x_n)$.
Определим предикат T^* - <u>расширенный предикат Клини</u>:

$$T^*(e,x,y,k)\Leftrightarrow\left\langle \begin{array}{c}\text{программа } \overline{\nu}e \text{ кончает работу над аргументом}\\ \text{x ровно за k шагов и выдает результат y.}\end{array}\right\rangle$$

или, более формально

$$T^*(e,x,y,k)\Leftrightarrow((t_f^*(x)=k)\&(P(x)=y)), \text{ где } P=\overline{\nu}e.$$

<u>Теорема</u>. Предикат T^* разрешим.

Доказательство.

(На основе тезиса Черча). По номеру e восстановим текст программы $\overline{\nu}e$. В качестве исходных данных зададим x и запустим программу $\overline{\nu}e$ на исполнение в режиме подсчета количества выполненных команд. Если программа закончит свою работу со значением счетчика выполненных команд меньше k, то значение предиката $T^*(e,x,y,k)$ равно false. Если значение счетчика выполненных команд достигнет k, то предикат $T^*(e,x,y,k)$ принимает значение true только в случае, когда программа $\overline{\nu}e$ на k шаге заканчивает свою работу и в нулевом регистре находится значение y. Во всех же других случаях достижения счетчиком выполненных команд значения k предикат $T^*(e,x,y,k)$ равен false.

Определение. Предикат $T(e, x, n) \stackrel{def}{=} T^*(e, x, l(n), r(n))$ будем называть предикатом Клини.

Теорема. (Клини о нормальной форме). Для любого алгоритма f выполняется равенство $f(x) \approx l(\mu_n T(\nu f, x, n))$.

Доказательство.

Пусть x и y произвольны. Имеем

$$f(x)=y \Leftrightarrow T^*(\nu f, x, y, t_f^*(x)) \Leftrightarrow$$
$$\Leftrightarrow T^*(\nu f, x, l(n), r(n)), \text{ где } n=c(y, t_f^*(x))$$
$$\Leftrightarrow \exists n(T^*(\nu f, x, l(n), r(n)) \& l(n)=y) \Leftrightarrow$$
$$\Leftrightarrow \exists n(T(\nu f, x, n) \& l(n)=y) \Leftrightarrow$$
$$\Leftrightarrow y=l(\mu_n T(\nu f, x, n)),$$

что и доказывает требуемое.

Универсальные алгоритмы.

Будем говорить, что алгоритм $U:N^{n+1} \to N$ является универсальным для вычислимых функций вида $f:N^n \to N$, если для любого f существует номер e, $e \in N$, такой что $f(x)=U(e,x)$.

Из теоремы Клини о нормальной форме вытекает существование универсального алгоритма, более того, для конкретной нумерации ν. В качестве универсального алгоритма можно взять следующий алгоритм U:

$$U(e,x)= l(\mu_n T(e, x, n)).$$

Выделим это утверждение как отдельную теорему

Теорема.(об универсальных алгоритмах) Существуют универсальные алгоритмы.

Замечание. Универсальный алгоритм задает некоторую нумерацию вычислимых функций (одной переменной). Функцию с номером e в этой нумерации будем обозначать

$$U_e: U_e(x)= U(e,x)$$

Этот же универсальный алгоритм задает нумерацию вычислимых функций с фиксированным числом аргументов; в такой нумерации k-местную функцию с номером e будем обозначать U_e^k.

Обозначение. $!U(e,x)$ – всюду определенный предикат, который принимает значение true в том и только в том случае, когда $U(e,x)\uparrow$

Алгоритмическая неразрешимость проблемы самоприменимости.

Существование алгоритмически неразрешимых предикатов очевидно из мощностных соображений: всего предикатов типа $N \to \{true, false\}$ континуум, а разрешимых предикатов не больше, чем алгоритмов, т. е. счетное число. Однако это соображение мало содержательно. Интересней то, что можно явно построить предикат, определяемый частичной вычислимой функцией, который является алгоритмически неразрешимым. Таким предикатом является предикат, выражающий свойство самоприменимости:

$$!U(x,x),$$

т. е. свойство “алгоритм с номером x применим к своему номеру x’.

Теорема.(алгоритмическая неразрешимость проблемы самоприменимости). Проблема самоприменимости алгоритмически неразрешима, т. е. невозможен полный алгоритм α, такой что

$$\forall x(\alpha(x)=0 \Leftrightarrow U(x,x)\uparrow). \qquad (*)$$

(Здесь используется определение: предикат разрешим, если его представляющая функция вычислима; всюду определенную функцию f(x) называют представляющей предиката R(x), если $\forall x(f(x)=0 \Leftrightarrow R(x))$.

Доказательство.

Положим $F(x)=\overline{sg}\,(U(x,x))$. Сначала докажем, что вычислимая функция F непродолжима до всюду определенной вычислимой функции, т.е. не существует всюду определенной вычислимой функции G, такой что

$$\forall x(F(x)\uparrow \Rightarrow F(x)=G(x)). \qquad (1)$$

Допустим, что имеется полный алгоритм G со свойством (1). Тогда

$$F(\nu G)\approx\overline{sg}\,(U(\nu G,\nu G))\approx\overline{sg}\,(G(\nu G)). \qquad (2)$$

В силу полноты алгоритма G имеем $\overline{sg}\,(G(\nu G))\uparrow$. Отсюда вытекает, что $F(\nu G)\uparrow$ и, следовательно, в силу (2)

$$F(\nu G)\approx\overline{sg}\,(G(\nu G)), \qquad (3)$$

а в силу (1) имеем $F(\nu G)=G(\nu G)$ – что противоречит (3).

Таким образом, непродолжимость F до полного алгоритма доказана. Теперь можно завершить доказательство теоремы.

Допустим, что существует полный алгоритм α со свойством (*). Рассмотрим функцию G, определяемую условием:

$$G(x)\approx \text{if } \alpha(x)=0 \text{ then } F(x) \text{ else } 0.$$

Поскольку α – полный алгоритм, то G – всюду определенная вычислимая функция. Очевидно, что она является продолжением F. Полученное противоречие завершает доказательство теоремы.

Отметим, что конструкция функции F является одной из реализаций идеи диагонали; среди других применений этой идеи – доказательство несчетности множества вещественных чисел. В определенном смысле, эта же идея лежит в основе многих логических парадоксов.

Задача 1. Докажите, что свойства $!U(x,x+7)$ и $!U(x,x^2)$ алгоритмически неразрешимы.

Задача 2. Пусть f – монотонно неубывающая вычислимая функция. Докажите, что свойство $!U(x,f(x))$ алгоритмически неразрешимо.

Задача 3. Для каких вычислимых f разрешимо или неразрешимо свойство $!U(f(x),x)$?

Задача 4. Постройте вычислимую функцию, не мажорируемую никакой полной вычислимой функцией.(Всюду определенная функция f мажорирует функцию g, если $\forall x : g(x)\uparrow \Rightarrow g(x)\le f(x)$).

Итерационная теорема (s-m-n-теорема) Клини.

Рассмотрим вычислимую функцию f от двух групп аргументов (например, просто от двух аргументов) x,y. Зафиксируем значения x, скажем пусть $x=x_0$. Функция $f(x_0,y)$ будет, очевидно, вычислимой функцией от y. (Почему?) Зададимся вопросом, как алгоритмически построить программу для вычисления этой функции по номеру f и по x_0. Этот вопрос сродни программистскому вопросу об универсальном способе осуществления частичного вычисления (соответствующий раздел теории программирования часто называют теорией частичных и смешанных вычислений). Некоторый общий ответ на обсуждаемый вопрос дает следующая теорема.

Теорема (итерационная теорема Клини). Пусть $e\in N$, $x\in N^m$, $y\in N^n$. Можно построить полный алгоритм $s:N^{m+1}\to N$, такой что для всех e, x, y

$$U(e,x,y)\approx U_{S(e,x,)}(y).$$

(В частности, для любой вычислимой функции f выполнено

$$f(x,y)\approx U(s(\nu f,x,),y) \text{ для всех } x,y.).$$

Доказательство(по тезису Черча).

Для наглядности считаем m=n=1. Для каждого фиксированного a через s(a) обозначим геделев номер программы Q_a, которая, исходя из данной начальной конфигурации

R0					
y	0	0	0	. . .	

, вычисляет f(a,y).

Пусть P – программа, вычисляющая функцию f. Тогда Q_a получается из P приписыванием спереди команд, преобразующих вышеуказанную конфигурацию в

R0	R1				
a	y	0	0	. . .	

Таким образом, определим Q_a, как следующую программу:

$$\begin{array}{l} T\ 0\ 1 \\ Z\ 0 \\ \text{a раз}\ \left\{\begin{array}{l} S\ 1 \\ \quad \vdots \\ S\ 1 \end{array}\right. \\ P[*] \end{array}$$

Теперь положим $s(a)=\nu(Q_a)$. Поскольку P фиксировано и ввиду эффективности нашей нумерации программ, функция s эффективно вычислима. Следовательно, по тезису Черча s – вычислимая функция. По построению $U_{s(a)}(y)\approx f(a,y)$ для каждого a.

Упражнение. Докажите теорему для произвольных m и n.

Алгоритмическая неразрешимость равенства нулю полного алгоритма.

Введем отношение эквивалентности на натуральных числах, которое соответствует равенству функций с данными номерами:

$$a\approx b \stackrel{def}{=} \forall x(U(a,x)=U(b,x)).$$

Обычно мы будем рассматривать функции фиксированной арности, так что отношение ≈ определяется для различных фиксированных чисел, определяющих количество компонент в x.

Обозначим посредством $TOTAL_n$ множество n-местных полных (всюду определенных) вычислимых функций. Посредством $zero_n$ будем обозначать n-местную функцию, тождественно равную 0:

$$zero_n(X) = 0.$$

Интуитивно правдоподобно, что нет эффективной процедуры, которая по программе или номеру полной вычислимой функции говорила бы. "равна ли она тождественно нулю, т.е. функции zero". Дело в том, что программа сама по себе мало информативна в части, касающейся поведения вычислимой функции в целом. И общим способом проверки равенства нулю является по-аргументная проверка этого свойства – а это требует перебора бесконечного числа ее аргументов.

<u>Теорема</u> (Алгоритмическая неразрешимость равенства нулю полного алгоритма)

$$\forall n(n\in TOTAL_m \Rightarrow !\alpha(n) \& (\alpha(n)=0 \Leftrightarrow n \approx \nu(zero_m))) \qquad (1)$$

для любого фиксированного m.

<u>Доказательство</u>. Мы рассмотрим случай m=1. Пусть F – алгоритм с неразрешимой проблемой применимости, т. е. такой, что невозможен полный алгоритм β, обладающий свойством:

$$\forall n(!\beta(n) \& (\beta(n)=0 \Leftrightarrow !F(n))). \qquad (2)$$

В качестве F можно взять, как мы показали ранее, функцию U(x,x), где U – универсальный алгоритм.

Рассмотрим следующую функцию

$$G(n,x) = \text{if } T(\nu F,n,x) \text{ then } 1 \text{ else } 0,$$

где $T(\nu F,n,x)$ предикат Клини. Она является полной вычислимой. Посмотрим на нее как на последовательность функций $G_n(x)=G(n,x)$. Нетрудно понять, что G_n есть тождественный ноль тогда и только тогда, когда $\neg!F(n)$. Поэтому распознаваемость равенства нулю G_n равносильно распознаваемости применимости F к n. А последнее невозможно. Проведем рассуждение более формально.

По итерационной теореме

$$G(n,x) = U(S(\nu G,n),x)$$

Рассмотрим функцию γ, определяемую равенством

$$\gamma(n) = S(\nu G, n).$$

Она является полной и вычислимой, а ее значениями являются номера полных вычислимых функций.

Допустим, что существует алгоритм α, обладающий свойством (1). Построим алгоритм ξ, удовлетворяющий равенству

$$\xi(n) = \overline{sg}(\alpha(\gamma(n))).$$

Этот алгоритм является полным. Теперь для произвольного n имеем

$$\xi(n) = 0 \Leftrightarrow \alpha(\gamma(n)) \neq 0 \Leftrightarrow \neg(\gamma(n) \approx \nu(zero)) \Leftrightarrow \exists x(G(n,x) \neq 0)$$

$$\Leftrightarrow \exists x T(\nu F,n,x) \Leftrightarrow !F(n).$$

Но полученное свойство противоречит (2).

<u>Упражнение</u> . Докажите теорему для произвольных m.

Задачи 1. Доказать, что невозможен алгоритм, который по номеру полной вычислимой ограниченной функции строит какую-либо ее верхнюю границу.

2. Доказать, что невозможен алгоритм, который по номеру периодической, полной вычислимой функции строит верхнюю границу длины ее периода.

3. Пусть f – вычислимая функция, доказать алгоритмическую неразрешимость свойств:

а) $|\mathrm{dom}\,(f)|=\infty$,

б) $|\mathrm{dom}\,(f)|<\infty$,

в) $|\mathrm{dom}\,(f)|=0$.

4.Пусть $f,g\in$ TOTAL, доказать алгоритмическую неразрешимость свойств:

а) $\forall x(f(x)\leq g(x))$,

б) $\forall x([f(x)]^2=g(x))$.

5. Доказать алгоритмическую неразрешимость свойства $|\mathrm{dom}\,(f)|=2$, где f – вычислимая функция с конечной областью определения.

6. Доказать алгоритмическую неразрешимость свойства $|\mathrm{dom}\,(f)|=|\mathrm{dom}\,(g)|$, где f,g– вычислимые функции, у которых $|\mathrm{dom}\,(f)|$, $|\mathrm{dom}\,(g)|\leq i$, $i\in N$.

7. Доказать алгоритмическую неразрешимость свойства $!U(x,0)$.

8. Доказать, что свойство $!U(k,x)$ может оказаться алгоритмически неразрешимым для некоторого k.

Понятие перечислимого множества.

Для доказательства алгоритмической неразрешимости равенства нулю полных вычислимых функций мы свели к этой задаче задачу об алгоритмической неразрешимости множества вида $\{x:!F(x)\}$. А среди таких множеств имеются алгоритмически неразрешимые. Понятие сводимости мы обсудим позже; сначала рассмотрим общие свойства множеств упомянутого вида. Они играют в теории алгоритмов заметную роль и, по сути дела, могут служить способом определения вычислимости. Кроме того, они достаточно точно соответствуют интуитивному понятию эффективно порождаемого множества.

Множество $S\subseteq N^k$ называется (рекурсивно или алгоритмически) перечислимым, если его можно представить в виде dom(f), где f – вычислимая функция. Предикат P типа $N^k\rightarrow\{true,\ false\}$ называется (рекурсивно или алгоритмически) перечислимым, если его множество истинности, т. е. множество $\{x\in N^k:P(x)\}$, перечислимо.

Очевидно, что всякое разрешимое множество является перечислимым и что обратное, вообще говоря, неверно (примером перечислимого, но не разрешимого множества является множество, соответствующее проблеме самоприменимости, т. е. $\{x : !U(x,x)\}$).

Другими, неочевидными примерами перечислимых, но не разрешимых множеств, являются: множество выводимых формул исчисления предикатов 1-го порядка; множество общезначимых формул логики предикатов 1-го порядка; множество выводимых формул формальной арифметики или теории множеств (в этих последних случаях истинные формулы уже не образуют перечислимых множеств); множество полиномов из $Z[x_1,\ldots,x_n]$, $n\geq 9$, обращающихся в ноль хотя бы на одном k-членном наборе целых чисел (неразрешимость 10-й проблемы Гильберта).

Простейшие свойства перечислимых множеств.

Перечислимости можно придать другой вид, как видно из следующего утверждения.

Утверждение 1. Множество $S\in N^k$ перечислимо $\Leftrightarrow$ существует вычислимая функция g, такая что $c_к(S)=g(N)$.

Доказательство. $\Rightarrow$. Пусть S=dom(f) для некоторого алгоритма f. Положим $g(n)=n\cdot sg(f(l_k^1(n),\ldots, l_k^k(n))+1)$. Эта функция, очевидно, обладает требуемыми свойствами.

$\Leftarrow$. Пусть $\underline{c}_к(S)=g(N)$, где g – вычислима. Положим

$$f(x)=\mu_n(c_к(x)=l(r(n))\&T(vg,l(n),r(n))),$$

где T – предикат Клини (из теоремы Клини о нормальной форме).

Функция g будет обладать требуемыми свойствами.

Способ перечисления можно канонизировать разными способами.

Утверждение 2. Всякое непустое перечислимое множество $S\in N^k$ перечислимо полным алгоритмом, т. е. имеется полный алгоритм g, такой что $\underline{c}_к(S)=g(N)$.

Доказательство. Пусть S=dom(f) и $x_0\in S$.

Определим g следующим образом:

$$g(0) = c_к(x_0),$$

$$g(n+1) = \begin{cases} g(n), \text{ если } \neg T(\nu f,\ c_k^{-1}(l(n)),r(n)) \\ l(n) \quad \text{в противном случае} \end{cases}, \text{ где } c_k^{-1}(m) = l_k^1(m),\ldots,l_k^k(m).$$

Нетрудно показать, что так построенная функция g обладает требуемыми свойствами.

Будем говорить, что функция $f : N\to N$ является стройной, если

$$\forall n(!f(n+1) \Rightarrow !f(n)).$$

Утверждение 3. Всякое перечислимое множество перечислимо стройным алгоритмом.

Доказательство. Пусть S=dom(f), f - алгоритм. Положим

$$g(0) = l(\mu_n(T(vf,\ c_k^{-1}\ (\underline{l}(n)),r(n))))$$

$$g(m+1) = l(\ \mu_n(g(m)\neq l(n)\ \&\ T(vf,\ c_k^{-1}(l(n)),r(n)))\ ).$$

Алгоритм g – стройный и перечисляет S : $c_к(S)=g(N)$.

Операции над перечислимыми и разрешимыми множествами.

Напомним обозначения для некоторых операций над множествами:

$\cup$ – объединение,

$\cap$ – пересечение,

co – дополнение (до соответствующего множества вида N^k),

pr_i – проекция вдоль i-й координаты

(для $S\in N^k$ и $1\le i\le k$ $pr_i(S)=\{(x_1,\ldots,x_{i-1},x_{i+1},\ldots,x_k) : \exists x_i(x_1,\ldots,x_k)\in S)\}$),

$\times$ – прямое произведение,

$t_{i,j}$ – перестановка (транспозиция) i-й и j-й координат.

Начнем с характеризации разрешимых множеств в терминах перечислимых.

Утверждение 4. Множество $S\in N^k$ разрешимо $\Leftrightarrow$ S и coS перечислимы.

Доказательство.$\Rightarrow$. Очевидно

$\Leftarrow$. Пусть S=dom(f) и coS=dom(g), f,g – алгоритмы. Положим

$$N(x)=\mu_n(T(vf,x,n) \vee T(vg,x,n)),$$

$$\alpha(x)= \text{if } T(vf,x,N(x)) \text{ then } 0 \text{ else } 1.$$

Поскольку $x\in S$ или $x\notin S$, то алгоритм N является полным. Теперь очевидно, что функция α является представляющей для S (и характеристической для coS).

Утверждение 5. Всякое перечислимое множество есть проекция разрешимого.

Доказательство. Пусть $S=dom(g)\in N^k$, g - алгоритм. Тогда по теореме о нормальной форме

$$X\in S\Leftrightarrow !\mu_n T(vg,X,n)$$

$$\Leftrightarrow \exists n T(vg,x,n)\Leftrightarrow x\in pr_{k+1}\{(x,n):T(vg,x,n)\}.$$

В силу разрешимости T эти эквивалентности дают требуемое.

Утверждение 6. Перечислимые множества замкнуты относительно $\cap$, $\cup$, $\times$, t, pr, но не замкнуты относительно co.

Разрешимые множества замкнуты относительно $\cap$, $\cup$, $\times$, t, co, но не замкнуты относительно pr.

Задача 1. Докажите утверждение 6.

Задача 2а. Докажите, что если f вычислима и строго возрастает, то f(N) – разрешимое множество.

Задача 2в. Докажите, что всякое бесконечное перечислимое множество содержит бесконечное разрешимое подмножество.

Вычислимость и перечислимость.

Посредством Гf будем обозначать график функции $f:N^k\rightarrow N$, т. е. множество $\{(x,y): f(x)=y\}$.

Утверждение 7. Для любой функции f

f вычислима $\Leftrightarrow$ Гf перечислим.

Доказательство. $\Rightarrow$. Пусть f вычислима. Тогда

$$f(X)=y\Leftrightarrow \exists m\, T^*(vf,X,y,m),$$

где T^* – предикат Клини, который, как мы знаем, разрешим. Отсюда вытекает, что график f есть проекция разрешимого множества.

$\Leftarrow$. Эта импликация непосредственно следует из следующего утверждения об “униформизации” – утверждения 8.

Утверждение 8. (Эффективная униформизация.) Для любого множества $S \in N^k$ можно построить вычислимую функцию $f{:}N^k \to N$, такую что

$$\Gamma f \subseteq S,\ \mathrm{dom} f = pr_{k+1} S.$$

Доказательство. Пусть $S=\mathrm{dom}(g)$, где g – алгоритм. Положим

$$f(X) = l(\mu_n(T(vg,X,l(n),r(n)))).$$

Задача 3. Докажите, что по всякой вычислимой функции f можно построить вычислимую функцию f^{-1}, обладающую свойствами:

$$!f(X) \Rightarrow\ !f^{-1}(f(X)),$$

$$!f^{-1}(y) \Rightarrow f(f^{-1}(y))=y.$$

Задача 4. Выясните перечислимость/разрешимость образа и прообраза перечислимого/разрешимого множества при вычислимом отображении.

Машина Тьюринга

Определение. Многоленточной машиной Тьюринга (МТ) называется семерка объектов:

$$MT=\langle Q,T,I,\delta,b,q_0,q_f\rangle,\ \text{где}$$

Q – конечное множество состояний управляющей головки;

T – конечное множество символов (алфавит) на лентах;

I – алфавит входной цепочки, $I \subseteq T$;

B – пустой символ, $B \in T \setminus I$;

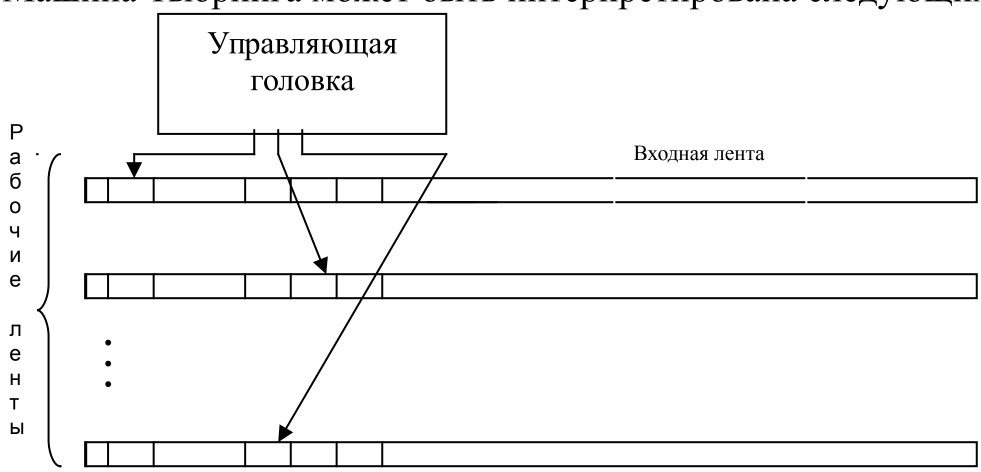
q_0 – начальное состояние;

q_f – заключительное(или допускающее) состояние;

δ – функция (частичная) перехода:

$$\delta{:}Q \times T^k \to Q \times (T \times \{L,R,S\})^k.$$

Машина Тьюринга может быть интерпретирована следующим образом:



Пусть $\delta(q,a_1,a_2,\ldots,a_k)=(q',(a'_1\, d_1),(a'_2\, d_2),,\ldots,(a'_k\, d_k))$ и МТ находится в состоянии q, а ее головка на i-ой ленте обозревает символ a_i, $1 \le i \le k$. Тогда за один шаг (такт) эта машина Тьюринга переходит в состояние q',заменяет i-ой ленте символ a_i на a'_i и сдвигает на этой же ленте головку в направлении d_k,$1 \le i \le k$.

Определим понятие *начальной конфигурации МТ* – головка находится в состоянии q_0, на входной ленте – входная цепочка, на остальных лентах – пустые символы. На всех лентах головка смотрит на первую позицию.

Цепочка из входных символов *допускается* машиной Тьюринга тогда и только тогда, когда МТ, начав работу в начальной конфигурации, сделав некоторую последовательность шагов, попадает в заключительное состояние.

Упражнение. Дайте определение языка, распознаваемого данной машиной Тьюринга.

Мгновенное описание (текущая конфигурация) k-ленточной машины Тьюринга определяется как набор $(\alpha_1,\dots,\alpha_k)$, где α_i для каждого i представляет собой слово xqy, причем xy – слово, а q текущее состояние машины. Головка на i-ой ленте обозревает символ, стоящий справа от q.

Если мгновенное описание α переходит в мгновенное описание β за один шаг МТ, то пишут $\alpha \xrightarrow[\text{МТ}]{} \beta$. Если $\alpha_1 \xrightarrow[\text{МТ}]{} \alpha_2 \xrightarrow[\text{МТ}]{} \dots \xrightarrow[\text{МТ}]{} \alpha_n$, то пишут $\alpha_1 \xrightarrow[\text{МТ}]{+} \alpha_n$. Если $\alpha_1=\alpha_n$, либо $\alpha_1 \xrightarrow[\text{МТ}]{+} \alpha_n$, то пишут $\alpha_1 \xrightarrow[\text{МТ}]{*} \alpha_n$.

Можно рассматривать МТ как задающую *вычислимую функцию* (частичную) $f:Z^n \to Z$. Числа кодируются на входной ленте в виде слов со специальным маркером #, отделяющим их друг от друга. Если МТ останавливается, имея на ленте, выделенной в качестве выходной, целое число y (значение функции), то полагают f(x)=y.

Можно также определить понятие *преобразователя*.

Временная сложность T(n) машины Тьюринга равна наибольшему числу шагов, сделанных ею при обработке входа длины n (для всех входов длины n). Если на каком-нибудь входе длины n машина Тьюринга не останавливается, то для этого n значение T(n) не определено.

Емкостная сложность S(n) машины Тьюринга равна наибольшему расстоянию от левого конца ленты, которое должна пройти головка при обработке входа длины n (для всех входов длины n). Если головка машина Тьюринга на какой-то ленте неопределенно долго движется вправо, то для этого n значение S(n) не определено.

Замечание. Наряду с определением машин Тьюринга, у которых функция перехода представляет собой однозначную частичную функцию, в теории также рассматриваются машины Тьюринга с неоднозначными функциями перехода. Если функция перехода однозначна, машина Тьюринга называется детерминированной, в противном случае – недетерминированной. Функционирование недетерминированной машины определяется следующим образом. В случае, если на некотором шаге МТ попадает в конфигурацию, для которой функция перехода определена неоднозначно, то считается, что, начиная с этого момента, запускаются все возможные варианты работы машины и если хотя бы один из них попадет в заключительную конфигурацию, то считается, что недетерминированная машина Тьюринга распознала входную цепочку.

Пример 1. Машина Тьюринга с одной лентой, распознающая язык $L=\{a^n b^n\}$.

$Q=\{q_0, q_1, q_2, q_3, q_4, q_5, q_6, q_7, q_f\}$, $T=\{a, b, *, \_\}$, $I=\{a,b\}$, $B=\_$,

q_0 – начальное состояние,

q_f – заключительное состояние,

$\delta: Q\times T \to Q\times(T \cup \{L,R,S\})$, определяется следующей таблицей:

$\delta(q_0,a)=(q_1,*)$ $\delta(q_3,\_)=(q_4,L)$ $\delta(q_5,b)=(q_6,L)$

$\delta(q_1,*)=(q_2,R)$ $\delta(q_4,b)=(q_5,\_)$ $\delta(q_6,b)=(q_6,L)$

$\delta(q_2,a)=(q_2,R)$ $\delta(q_5,\_)=(q_5,L)$ $\delta(q_6,a)=(q_6,L)$

$\delta(q_2,b)=(q_3,R)$ $\delta(q_5,*)=(q_f,\_)$ $\delta(q_6,*)=(q_7,\_)$

$\delta(q_3,b)=(q_3,R)$ $\delta(q_7,\_)=(q_0,R)$.

Комментарий. В процессе работы машина Тьюринга стирает крайние символы входной цепочки. В начальном состоянии первый символ a на входной ленте заменяется символом *. Затем головка двигается вправо до символа _ в конце входной цепочки (состояния – q_1, q_2, q_3, q_4). Далее стирается символ b, расположенный в конце входной цепочки (состояния – q_5). Если в состоянии q_5 перед записанным символом _ расположен символ *, то МТ стирает его и переходит в заключительное состояние q_f. Если же в состоянии q_5 перед записанным символом _ расположен символ, отличный от *, то головка возвращается к символу * (состояние – q_6), стирает его(состояние –q_7) и переходит в начальное состояние для применения описанного процесса к более короткой цепочке. Машина Тьюринга попадает в неопределенное состояние, если а) в состоянии q_0 встретится символ b; б) в состоянии q_2 не будет найден символ b; в) в состоянии q_3 встретится символ a.

Заметим, что временная вычислительная сложность предложенной машины имеет порядок $O(n^2)$, а емкостная – $O(n)$, где n – длина входной цепочки.

Пример 2. Машина Тьюринга с двумя лентами, распознающая “правильную структуру” арифметического выражения.

Определение. Цепочку x, состоящую из символов [,] будем называть правильной скобочной структурой арифметического выражения, если

1. общее число во всей цепочке x открывающихся скобок совпадает с числом закрывающихся скобок в x.
2. в любом префиксе цепочки x количество открывающихся скобок не меньше числа закрывающихся.

Проверку этих двух условий обеспечивает следующая машина Тьюринга с двумя лентами:

$Q = \{q_0,q_1,q_2,q_3\}$;

$T = \{[,],\_,Z_0,Z_1\}$

$I = \{[,]\}$

$b=\_$

q_0 – начальное состояние;

q_3– заключительное состояние;

$\delta(q_0, [, \_) = (q_1, ([,R), (Z_0,R))$

$\delta(q_1, [, \_) = (q_1, ([,R), (Z_1,R))$

$\delta(q_1,], \_) = (q_2, (],S), (\_,L))$

$\delta(q_2,], Z_1) = (q_1, (],R), (\_,S))$

$\delta(q_2,], Z_0) = (q_0, (],R), (\_,L))$

$\delta(q_0, \_, \_) = (q_3, (\_,S), (\_,S))$

Комментарий. Фактически предлагаемая машина Тьюринга на рабочей ленте в единичной системе считает разность между открывающимися и закрывающимися скобками. При этом первую открывающуюся скобку помечает символом Z_0, а остальные Z_1. Эти символы стираются с рабочей ленты при чтении закрывающихся скобок на входной ленте. МТ переходит в начальное состояние при стирании Z_0, которое обеспечивает переход в заключительное состояние при чтении на входной ленте символа _. Если в префиксе входной цепочки число закрывающихся скобок превысит число открывающихся, то МТ “сломается” по попытке сдвинуться влево за начальный маркер рабочей ленты. Если входная цепочка представляет собой префикс правильной скобочной структуры, то предлагаемая машина Тьюринга попадет в конфигурацию, для которой функция перехода не определена (ситуация $\delta(q_1, \_, \_)$).

Заметим, что как временная, так и емкостная вычислительные сложности предложенной машины имеет порядок O(n), где n – длина входной цепочки.

Упражнение. Постройте одноленточную машину Тьюринга, распознающую “правильную структуру” арифметического выражения.

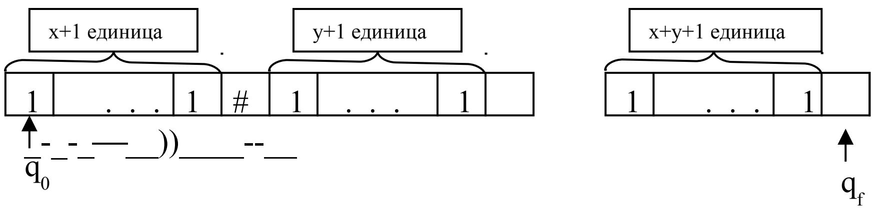
Определение. Машины Тьюринга будем называть эквивалентными, если они распознают один и тот же язык.

Можно доказать следующую теорему:

Теорема. Для любой многоленточной машины Тьюринга можно построить ей эквивалентную одноленточную машину Тьюринга.[*]

Пример. Одноленточная машина Тьюринга, вычисляющая функцию $f:N\times N\to N$ такую, что $\forall x,y\in N$ $f(x,y)=x+y$. Считаем, что x и y представлены в единичной системе: $0\leftrightarrow 1$, $1\leftrightarrow 11$, $2\leftrightarrow 111,\ldots$ и разделены символом #.

Начальная конфигурация: Заключительная конфигурация:



Функция перехода:

$(q_0,1)\to(q_0,R)$ $(q_0,\#)\to(q_1,1)$ $(q_1,1)\to(q_1,R)$

$(q_1,\_)\to(q_2,L)$ $(q_2,1)\to(q_3,L)$ $(q_3,1)\to(q_f,\_)$.

Упражнение. Постройте одноленточные машины Тьюринга, вычисляющие следующие функции:

а) $h:N\to N$, $h(x)=\dot{-}(x,1)=x\dot{-}1=\begin{cases}0, & \text{если } x=0\\ x-1, & \text{если } x>0\end{cases}$;

б) $h:N\to N$, $h(x)=-(x,1)=x-1=\begin{cases}\text{не определена}, & \text{если } x=0\\ x-1, & \text{если } x>0\end{cases}$;

в) $h:N^2\to N$, $h(x)=x\dot{-}y=\begin{cases}0, & \text{если } x<y\\ x-y, & \text{если } x\geq y\end{cases}$;

г) $h:N^2\to N$, $h(x)=x-y=\begin{cases}\text{не определена}, & \text{если } x<y\\ x-y, & \text{если } x\geq y\end{cases}$;

д) $h:N\to N$, $h(x)=sg(x)=\begin{cases}0, & \text{если } x=0\\ 1, & \text{если } x\neq 0\end{cases}$;

е) $h:N\to N$, $h(x)=\overline{sg}(x)=\begin{cases}1, & \text{если } x=0\\ 0, & \text{если } x\neq 0\end{cases}$;

ж) $h:N^2\to N$, $h(x,y)=|x-y|$;

з) $h:N^2\to N$, $h(x,y)=\min\{x-y\}$.

---

[*] Гладкий А. В. Формальные грамматики и языки. Наука, 1973

и) $h:N\to N$, $h(x)=odd\,(x)=\begin{cases}1, & \text{если } x \text{ - нечетное}\\ 0, & \text{если } x \text{ - четное}\end{cases}$.

Определение. Частичную функцию $f:N\times N\to N^k$ будем называть *вычислимой по Тьюрингу*, если существует машина Тьюринга, вычисляющая ее.

В теории машин Тьюринга доказывается следующая теорема:

Теорема. Любая частично рекурсивная функция вычислима на одноленточной машине Тьюринга, и обратно*.

Упражнение. Дайте определения перечислимых и разрешимых множеств и предикатов в терминах машины Тьюринга.

Следствие. Проблема останова машин Тьюринга общего вида алгоритмически неразрешима.

Нормальные алгорифмы Маркова

Лучшее обоснование этого подхода к определению алгоритма приведено: А. А. Марков, Н. М. Нагорный. Теория алгорифмов. М. Наука 1984.

$A=\{a_1,\ldots a_n\}$, $B=\{\alpha_1,\ldots,\alpha_k,a_1,\ldots a_n\}$, $\alpha_1,\ldots,\alpha_k$ – вспомогательные символы.

X,X_1,X_2,Y,P,Q – цепочки над B, обозначение $X,X_1,X_2,Y,P,Q\in B^*$

λ - пустая цепочка, $\lambda\in A^*$

$P\to Q$ продукция

Применение продукции: если $X=X_1PX_2$ и $Y=X_1QX_2$, то говорят, что Y получено из X применением продукции $P\to Q$, обозначается $X\Rightarrow Y$.

Алгорифм Маркова U:

заданы алфавиты A и B, $A\subset B$;

простое правило $P\to Q$ $P,Q\in B^*$;

заключительное правило $P\to\bullet Q$ $P,Q\in B^*$;

схема правил:

$$\begin{cases}1.P_1\to(\bullet)Q_1\\ 2.P_2\to(\bullet)Q_2\\ \quad\ldots\\ n.P_n\to(\bullet)Q_n\end{cases}$$

{(•) – обозначает, что в этом месте символ • может как находиться, так и отсутствовать.}

Одиночное преобразование цепочки по алгорифму Маркова (шаг работы алгорифма) представляет собой выбор первого применимого правила с наименьшим номером и применение его к первому слева вхождению $P_{\text{выбранное}}$, в цепочку, путем замены этого вхождения на $Q_{\text{выбранное}}$. Применение алгорифма Маркова к входной цепочке представляет собой последовательное применение одиночных правил, заканчивающееся первым применением заключительного правила. Цепочка, полученная применением алгорифма Маркова к входной цепочке, называется результатом работы алгорифма

* А.. И. Мальцев. Алгоритмы и рекурсивные функции, Наука. М., 1965.

Маркова к данной входной цепочке. Количество шагов работы алгорифма Маркова к данной входной цепочке называется вычислительной сложностью алгорифма Маркова к данному входу.

Обозначение:

$U: R \Rightarrow R_1$ исполнение одного шага алгорифма Маркова к цепочке R, результат R_1.

$U: R \supset$ невозможность применения правил к цепочке R.

$U: R \Rightarrow_* R_n$ получение цепочки R_n из R за несколько шагов.

$U: R \Rightarrow \bullet R`$ применение заключительного правила после нескольких шагов.

$U: R \supset \bullet$ невозможность нормального вывода, т. е. либо алгорифм Маркова работает вечно, либо после применения нескольких простых шагов наступает ситуация невозможности применения никакого правила алгорифма Маркова.

Примеры:

1.пусть A={b,c},

$$\begin{cases} 1.b \to \bullet\lambda \\ 2.c \to c \end{cases}$$

U перерабатывает первое вхождение буквы b в слово без этого b. Если в слове нет b, то U неприменим. $U(\lambda)=\lambda$.

2.$\forall X \in A^* \ U(X)=\lambda$

$$\begin{cases} 1.a_1 \to \lambda \\ \dots \\ n.a_n \to \lambda \\ n+1.\lambda \to \bullet\lambda \end{cases}$$

Краткая запись

$\xi \to \lambda \quad (\xi \in A)$

3. .$\forall X \in A^* \ U(X)=Q,\ Q \in A^*$

1.$\xi \to \lambda \quad (\xi \in A)$

2.$\lambda \to \bullet Q$

4. .$\forall X \in A^* \ U(X)=XQ,\ Q \in A^*$

1. $\alpha\xi \to \xi\alpha \quad (\xi \in A)$
2. $\alpha \to \bullet Q$
3.$\lambda \to \alpha$

5. $U(a_{j_0}\ a_{j_1} \dots a_{j_k}) = a_{j_1} \dots a_{j_k} \qquad U(\lambda) = \lambda$

1. $\alpha\xi \to \bullet\lambda \quad \xi \in A$
2. $\alpha \to \bullet\lambda$
3.$\lambda \to \alpha$

Неверное решение!

1.$\xi \to \bullet\lambda \quad \xi \in A$

почему ?

6. $U(a_{j_0}\ a_{j_1} \dots a_{j_k}) = a_{j_0} \dots a_{j_{k-1}} \qquad U(\lambda) = \lambda$

1. $\alpha\xi \to \xi\alpha \quad \xi \in A$
2. $\xi\alpha \to \bullet\lambda \quad \xi \in A$
3. $\alpha \to \bullet\lambda$

4. $\lambda\to\alpha$

7. $U(a_{j_0}\, a_{j_1} \ldots a_{j_k}) = a_{j_1} \ldots a_{j_k}\, a_{j_0}$ $\qquad U(\lambda) = \lambda$

1. $\alpha\xi\eta\to\eta\alpha\xi \quad \eta,\xi\in A$
2. $\alpha\xi\to\bullet\xi \quad \xi\in A$
3. $\alpha\to\bullet\lambda$
4. $\lambda\to\alpha$

8. $U(a_{j_0}\, a_{j_1} \ldots a_{j_k}) = a_{j_0}\, a_{j_1} \ldots a_{j_k}\, a_{j_0}$ $\qquad U(\lambda) = \lambda$

1. $\alpha\xi\eta\to\eta\alpha\xi \quad \eta,\xi\in A$
2. $\alpha\xi\to\bullet\xi \quad \xi\in A$
3. $\beta\xi\to\alpha\xi\xi \quad \xi\in A$
4. $\beta\to\bullet\lambda$
5. $\lambda\to\beta$

Пример

$abc\Rightarrow^5\beta abc\Rightarrow^3\alpha aabc\Rightarrow^1 a\alpha abc\Rightarrow^1$
$ab\alpha ac\Rightarrow^1 abc\alpha a\Rightarrow^2 abca$

9. $U(a_{j_0}\, a_{j_1} \ldots a_{j_k}) = a_{j_k} \ldots a_{j_1}\, a_{j_0}$ $\qquad U(\lambda) = \lambda$

1. $\alpha\xi\eta\to\eta\alpha\xi \quad \eta,\xi\in A$
2. $\beta\xi\to\beta\alpha\xi \quad \xi\in A$
3. $\beta\alpha\xi\to\xi\beta \quad \xi\in A$
4. $\beta\to\bullet\lambda$
5. $\lambda\to\beta$

Пример

$abc\Rightarrow^5\beta abc\Rightarrow^2\beta\alpha\ abc\Rightarrow^1\beta b\alpha ac\Rightarrow^1$
$\beta bc\alpha a\Rightarrow^2\beta\alpha bc\alpha a\Rightarrow^1\beta c\alpha b\alpha a\Rightarrow^2$
$\beta\alpha c\alpha b\alpha a\Rightarrow^3 c\beta\alpha b\alpha a\Rightarrow^3 cb\beta\alpha a\Rightarrow^3$
$cba\beta\Rightarrow^4 cba$

10. $U(a_{j_0}\, a_{j_1} \ldots a_{j_k}) = a_{j_0}\, a_{j_1} \ldots a_{j_k}\, a_{j_k} \ldots a_{j_1}\, a_{j_0}$ $\qquad U(\lambda) = \lambda$

1. $\alpha\xi\eta\to\eta\alpha\xi \quad \eta,\xi\in A$
2. $\beta\xi\eta\to\xi\beta\alpha\eta\eta \quad \eta,\xi\in A$
3. $\beta\xi\alpha\to\xi\beta \quad \xi\in A$
4. $\beta\to\bullet\lambda$
5. $\gamma\xi\to\beta\alpha\xi\xi \quad \xi\in A$
6. $\gamma\to\bullet\lambda$
7. $\lambda\to\gamma$

пример

$abc\Rightarrow^7\gamma abc\Rightarrow^5\beta\alpha aabc\Rightarrow^1\beta a\alpha abc\Rightarrow^1$
$\beta ab\alpha ac\Rightarrow^1\beta abc\alpha a\Rightarrow^2 a\beta\alpha bbc\alpha a\Rightarrow^1$
$a\beta b\alpha bc\alpha a^1\Rightarrow a\beta bc\alpha b\alpha a\Rightarrow^2 b\beta cc\alpha b\alpha a\Rightarrow^1$
$abc\beta c\alpha b\alpha a\Rightarrow^3 abcc\beta b\alpha a\Rightarrow^3\ abccb\beta a\Rightarrow^4$
$abccba$

Какова вычислительная сложность алгоритма ?

11. $U(a_{j_0}\, a_{j_1} \ldots a_{j_k}) = a_{j_0}\, a_{j_1} \ldots a_{j_k}\, a_{j_0}\, a_{j_1} \ldots a_{j_k}$ $\qquad U(\lambda) = \lambda$

1. $\xi\beta\gamma\gamma\to\gamma\gamma\xi \quad \xi,\eta\in A$
2. $\xi\beta\eta\to\eta\xi\beta \quad \xi,\eta\in A$
3. $\xi\alpha\to\alpha\xi\xi\beta \quad \xi\in A$
4. $\gamma\gamma\to\lambda$
5. $\alpha\to\bullet\lambda$
6. $\gamma\xi\to\xi\gamma \quad \xi\in A$
7. $\xi\gamma\to\alpha\xi\gamma\gamma\xi \quad \xi\in A$
8. $\lambda\to\gamma$

Пример

$abc\Rightarrow^8\gamma abc\Rightarrow^6 a\gamma bc\Rightarrow^6 ab\gamma c\Rightarrow^6 abc\gamma\Rightarrow^7$
$ab\alpha c\gamma\gamma c\Rightarrow^3 a\alpha bb\beta c\gamma\gamma c\Rightarrow^2 a\alpha bcb\beta\gamma\gamma c\Rightarrow^1$
$a\alpha bc\gamma\gamma bc\Rightarrow^3\alpha aa\beta bc\gamma\gamma bc\Rightarrow^2\alpha aba\beta c\gamma\gamma bc\Rightarrow^2$
$\alpha abca\beta\gamma\gamma bc\Rightarrow^1\alpha abc\gamma\gamma abc\Rightarrow^4\alpha abcabc\Rightarrow^5$
$abcabc$

Какова вычислительная сложность алгоритма ?

12. $U(a_{j_0}\, a_{j_1} \ldots a_{j_k}) = a_{j_0}\, a_{j_1} \ldots a_{j_k}\, Q$

1. $\xi\beta\gamma\gamma\to\gamma\gamma\xi \quad \xi,\eta\in A$
2. $\xi\beta\eta\to\eta\xi\beta \quad \xi,\eta\in A$
3. $\xi\alpha\to\alpha\xi\xi\beta \quad \xi\in A$

Пример

$abc\Rightarrow^8\gamma abc\Rightarrow^6 a\gamma bc\Rightarrow^6 ab\gamma c\Rightarrow^6 abc\gamma\Rightarrow^7$
$ab\alpha c\gamma\gamma c\Rightarrow^3 a\alpha bb\beta c\gamma\gamma c\Rightarrow^2 a\alpha bcb\beta\gamma\gamma c\Rightarrow^1$
$a\alpha bc\gamma\gamma bc\Rightarrow^3\alpha aa\beta bc\gamma\gamma bc\Rightarrow^2\alpha aba\beta c\gamma\gamma bc\Rightarrow^2$
$\alpha abca\beta\gamma\gamma bc\Rightarrow^1\alpha abc\gamma\gamma abc\Rightarrow^4\alpha abcQabc\Rightarrow^5$
$abcQabc$

Какова вычислительная сложность алгоритма ?

4. $\gamma\gamma\to Q$
5. $\alpha\to\bullet\lambda$
6. $\gamma\xi\to\xi\gamma$ $\xi\in A$
7. $\xi\gamma\to\alpha\xi\gamma\gamma\xi$ $\xi\in A$
8. $\lambda\to\gamma$

13.A={1}, x- натуральное число в унарной системе

$U(x) = x \text{ div } 5$ $U(\lambda) = \lambda$

1. $\alpha 11111\to 1\alpha$
2. $\alpha 1111\to\bullet\lambda$
3. $\alpha 111\to\bullet\lambda$
4 $\alpha 11\to\bullet\lambda$
5 $\alpha 1\to\bullet\lambda$
6 $\alpha\to\bullet\lambda$
7. $\lambda\to\alpha$

14.A={1}, x- натуральное число в унарной системе

$U(x) = 5x$ $U(\lambda) = \lambda$

1. $\alpha 1\to 11111\alpha$
2 $\alpha\to\bullet\lambda$
3. $\lambda\to\alpha$

15.A={1,0}, x- натуральное число в унарной системе

U(x) = {представление x в двоичной системе} $U(\lambda) = 0$

1.$\beta\gamma\xi\to\xi\beta\gamma$ $\xi\in A$
2.$\delta 11\to 1\delta$
3.$\delta 1\beta\to\beta 1$
4.$\delta\beta\to\beta 0$
5.$\alpha\xi\to\alpha\delta\xi$ $\xi\in A$
6.$\alpha\beta\to\lambda$
7.$\gamma\to\bullet\lambda$
8.$\lambda\to\alpha\beta\gamma$

Системы Поста.

Элементарная операция (продукция):

$$x_0S_1x_1S_2\ldots\quad x_{m-1}S_mx_m\to y_0S_{i_1}y_1S_{i_2}\ldots S_{i_n}y_n,$$

где S_i – произвольная цепочка над алфавитом A (переменная).

x_i,y_j – фиксированные цепочки над A, при этом $i_1,\ldots i_n$ выбираются из $1,\ldots,m$ и могут быть одинаковыми.

Пример. A={a,b} $aS_1bS_2\to S_2aS_2a$

$\underset{S_1,S_2=\lambda}{ab\Rightarrow aa}$ $\underset{S_1=\lambda,\ S_2=a}{aba\Rightarrow aaaa}$ $\underset{S_1=\lambda}{abba\Rightarrow baabaa}$ $\underset{S_1=b}{abba\Rightarrow aaaa}$

Задача A={a,b} $S_1bS_2aaS_3b\to S_3abS_1$

babaabbaab

$\Rightarrow$ bbaaab $\Rightarrow$ bbaaabba

$S_1=\lambda,\ S_2=ab,\ S_3=bbaa$ $\quad$ $S_1=ba,\ S_2=\lambda,,\ S_3=bbaa$

$\Rightarrow$ abbabaa $\quad$ $\Rightarrow$ bbaaab

$S_1=babaa,\ S_2=b,\ S_3=\lambda$ $\quad$ $S_1=babaab,\ S_2=\lambda,,\ S_3=\lambda,$

Определение. Каноническая система Поста J это
Алфавит A, множество аксиом $E\subset A^*$, множество продукций P

$$L(J)=\{x\in A^*:E \Rightarrow_J x\}$$

Пример. $A=\{a,b\}$, $E=\{\lambda,a,b\}$, $P=\{S\rightarrow aSa,\ S\rightarrow bSb\}$

$$L(J)=\{\text{палиндромы над } A\}$$

Определение. X порождаемо по Посту, если существует $J=\langle B,E,P\rangle$, $B\supseteq A$ такое, что $X=L(J)\cap A^*$.

Определение. J нормальная, если в P правила вида $xS\rightarrow Sy$.

Теорема. (Э. Пост)Всякая каноническая система может быть порождена нормальной системой.

(Доказательство, например, М. Минский, Вычисления и автоматы, М. Мир, 1971).

Задача 1. Построить каноническую систему Поста, порождающую множество нечетных натуральных чисел. Числа представлены в унарной системе: $0\leftrightarrow\lambda$, $1\leftrightarrow 1$, $2\leftrightarrow 11$, $3\leftrightarrow 111$, … .

Решение: $A=\{1\}$, $B=A$, переменные x, аксиома 1; $P=\{x\rightarrow x11\}$

Задача 2. Построить каноническую систему, порождающую арифметические тождества положительных целых чисел с одной операцией сложения. Числа представлены в унарной системе.
Пример 111+11111=11111111.

Решение:$A=\{1,+,=\}$, $B=A$, переменные x, y, z;

аксиомы: $\{1+1=11\}$; $P=\{x+y=z\rightarrow x1+y=z1,\ x+y=z\rightarrow x+y1=z1\}$

Задача 3. Построить каноническую систему, порождающую арифметические тождества положительных целых чисел с одной операцией умножения. Числа представлены в унарной системе.
Пример 111*11=111111.

Решение:$A=\{1,*,=\}$, $B=A$, переменные x, y, z;

аксиомы: $\{1*1=1\}$; $P=\{x*y=z\rightarrow x1*y=zy,\ x*y=z\rightarrow y*x=z\}$

Задача 4. Построить каноническую систему Поста, порождающую множество квадратов натуральных чисел. Числа представлены в унарной системе.

Решение:. $A=\{1\}$, $B=A\cup\{\alpha,\}$, переменные x, y, аксиома 1α;

$P=\{x\alpha y\rightarrow x11\alpha yx,\ x\alpha y\rightarrow y\}$

Алгоритм основан на тождестве $(n+1)^2 = n^2+2n+1$, переменной x соответствуют нечетные числа вида 2n+1, а переменной y – квадрат предыдущего числа n.

Задача 5. Построить каноническую систему, порождающую множество правильно сформированных слов из скобок типа

( ), (( )), (( ) ( )), (( )) ( ), ((( )( )) ((( )))),

в которых каждой левой скобке должна соответствовать своя правая скобка.

Решение:

. A={(,)},, переменные x, y, аксиомы {( )}

P={x→(x), x→xx, x( )y→xy}

<u>Задача 6</u>. Какой класс скобочных структур будет порождать каноническая система в задаче 5, если все ее продукции заменить единственной продукцией: xy→x( )y.

<u>Задача 7</u>. Построить каноническую систему, порождающую множество положительных целых чисел, не являющихся простыми. Числа представлены в унарной системе.

Решение:A={1}, B=A∪{α,β,γ}, переменные x,y,z; аксиома α;

P={αx→αx1, αx→xβγ, xβyγz→ xβy1γxz, 11xβy11γz→z}

(1) (2) (3) (4)

Правило (1) служит для набора x единиц, затем с помощью правила (2) размечается место для второго множителя y и результата z – произведения x на y; правило (3) служит для построения тождества x*y=z l для любых x и y. Применение правила (4) приводит к получению составного числа в качестве результата канонической системы:

k единиц

$\alpha \Rightarrow_1 \alpha 1 \Rightarrow_1 \alpha 11 \quad \Rightarrow_1$ k раз $\alpha 11\ldots 1$

$_2\Downarrow \quad _2\Downarrow \quad _2\Downarrow$

$1\beta\gamma_3 \quad 11\beta\gamma \quad 11\ldots 1\beta$

$\Downarrow_3 \quad _3\Downarrow$ n- раз

$\beta 1\gamma 1 \quad 11\beta 1\gamma 11 \quad _3\Downarrow$

$_3\Downarrow$.

$11\beta 11\gamma 1111 \Rightarrow_3 11\beta 111\gamma 111111$ n-раз$\Rightarrow_3$.

$_4\Downarrow \quad _4\Downarrow \quad _4\Downarrow$.

1111 111111 1111…1111 2* n единиц и

$_4\Downarrow$

k*n единиц

Задача 8. Построить систему Поста, порождающую множество простых чисел. Числа представлены в унарной системе.

Решение:A={1}, B=A∪{α,β,γ,δ}, переменные x,y,z; E={ α111, 11}

(1) (2) (3) (4) (5) (6)

P={αx→αx1, αx1→γxδβx1, x1δy1→ 1xδ1y, xγδy1→γxδy1, xγy1δzβ→γxyδβz, 11γδzβ1→z1}

Алгоритм основан на идеях решета Эратосфена. Правило (1) служит для набора x единиц, затем с помощью правила (2) запускается процесс проверки x на простоту – правила (3), (4), (5). Он успешный, если возникает возможность применения правила (6),которое приводит к получению простого числа в качестве результата. Если x составное, то возникает цепочка, к которой нельзя применить ни одно правило – вспомогательные символы γ,δ склеены, символ β в конце цепочки:

$\alpha 111 \quad \Rightarrow_1 \quad \alpha 1111 \quad \Rightarrow_1 \quad \alpha 11111$

$_2\Downarrow \quad \Rightarrow_6 \; 111 \quad _2\Downarrow \quad _2\Downarrow$

$\gamma 11\delta\beta 11 \quad \gamma 111\delta\beta 1111 \quad \gamma 1111\delta\beta 11111$

$_3\Downarrow \quad _3\Downarrow \quad _3\Downarrow$

$1\gamma 1\delta 1\beta 11 \quad 1\gamma 11\delta 1\beta 111 \quad 1\gamma 111\delta 1\beta 1111$

$_3\Downarrow \quad _3\Downarrow \quad _3\Downarrow$

$11\gamma\delta 11\beta 1 \quad 11\gamma 1\delta 11\beta 11 \quad 11\gamma 11\delta 11\beta 111$

${}_4\Downarrow$
$\gamma11\delta11\beta1$
${}_3\Downarrow$
$1\gamma1\delta111\beta$
${}_5\Downarrow$
$\gamma1\delta\beta111$
${}_3\Downarrow$
$1\gamma\delta1\beta11$
${}_4\Downarrow$
$\gamma1\delta1\beta11$
${}_3\Downarrow$
$1\gamma\delta11\beta1$
${}_4\Downarrow$
$\gamma1\delta11\beta1$
${}_3\Downarrow$
$1\gamma\delta111\beta$
конец

${}_3\Downarrow$
$111\gamma\delta111\beta1$
${}_4\Downarrow$
$\gamma111\delta111\beta1$
${}_3\Downarrow$
$1\gamma11\delta1111\beta$
${}_5\Downarrow$
$\gamma11\delta\beta1111$
${}_3\Downarrow$
$1\gamma1\delta1\beta111$
${}_3\Downarrow$
$11\gamma\delta11\beta11$
${}_4\Downarrow$
$\gamma11\delta11\beta11$
${}_3\Downarrow$
$1\gamma1\delta111\beta1$
${}_3\Downarrow$
$11\gamma\delta1111\beta$
конец

${}_3\Downarrow$
$111\gamma1\delta111\beta11$
${}_3\Downarrow$
$1111\gamma\delta1111\beta1$
${}_4\Downarrow$
$\gamma1111\delta1111\beta1$
${}_3\Downarrow$
$1\gamma111\delta11111\beta$
${}_5\Downarrow$
$\gamma111\delta\beta11111$
${}_3\Downarrow$
$1\gamma11\delta1\beta1111$
${}_4\Downarrow$
$11\gamma1\delta11\beta111$
${}_3\Downarrow$
$111\gamma\delta111\beta11$
${}_4\Downarrow$
$\gamma111\delta111\beta11$
${}_3\Downarrow$
$1\gamma11\delta1111\beta1$
${}_3\Downarrow$
$11\gamma1\delta11111\beta$
${}_5\Downarrow$
$\gamma11\delta\beta11111$
${}_3\Downarrow$
$1\gamma1\delta1\beta1111$
${}_3\Downarrow$
$11\gamma\delta11\beta111$
${}_3\Downarrow$
$\gamma11\delta11\beta111$
${}_3\Downarrow$
$1\gamma1\delta111\beta11 \quad \Rightarrow_6 11111$
${}_3\Downarrow$
$11\gamma\delta1111\beta1$
${}_4\Downarrow$
$\gamma11\delta1111\beta1$
${}_3\Downarrow$
$1\gamma1\delta11111\beta$
и т. д.

ЛИТЕРАТУРА

1. И. А. Лавров, Л. Л. Максимова. Задачи по теории множеств, математической логике и теории алгоритмов. Наука, М., 1975.
2. Н. Катленд. Вычислимость. Введение в теорию рекурсивных функций. Мир, М., 1983.
3. А. И. Мальцев. Алгоритмы и рекурсивные функции. Наука, М. 1965.
4. Э. Мендельсон. Введение в математическую логику. Наука. М. 1971.
5. С. К. Клини. Введение в метаматематику. ИЛ. М. 1957.
6. С. К. Клини. Математическая логика. Мир, М., 1973.
7. Х. Роджерс. Теория рекурсивных функций и эффективная вычислимость. Мир, М., 1972.
8. А. О. Слисенко. Основы теории алгоритмов (методическое пособие) ЛИИАН СССР, Л. 1991.