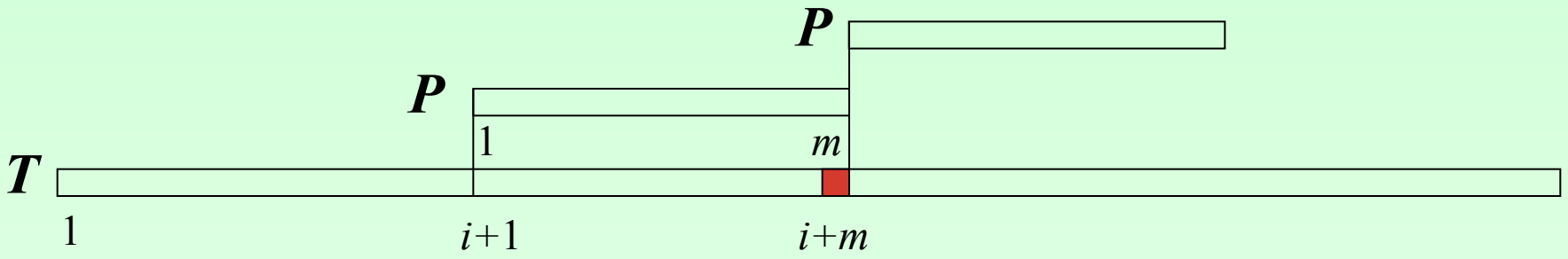
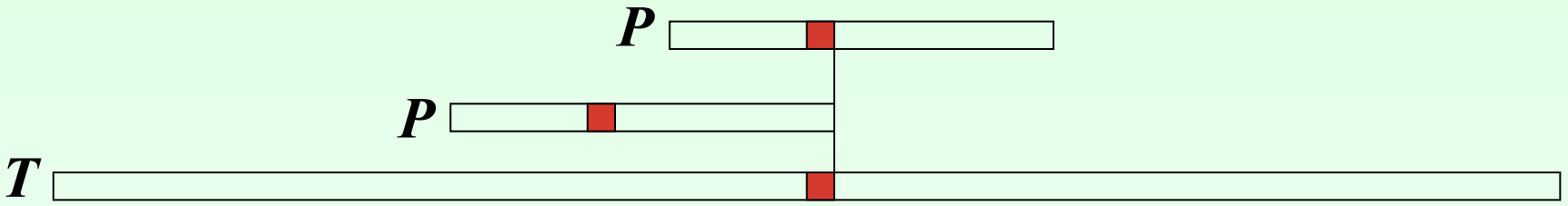
Алгоритм Бойера-Мура

Сравнение символов – справа налево !!!

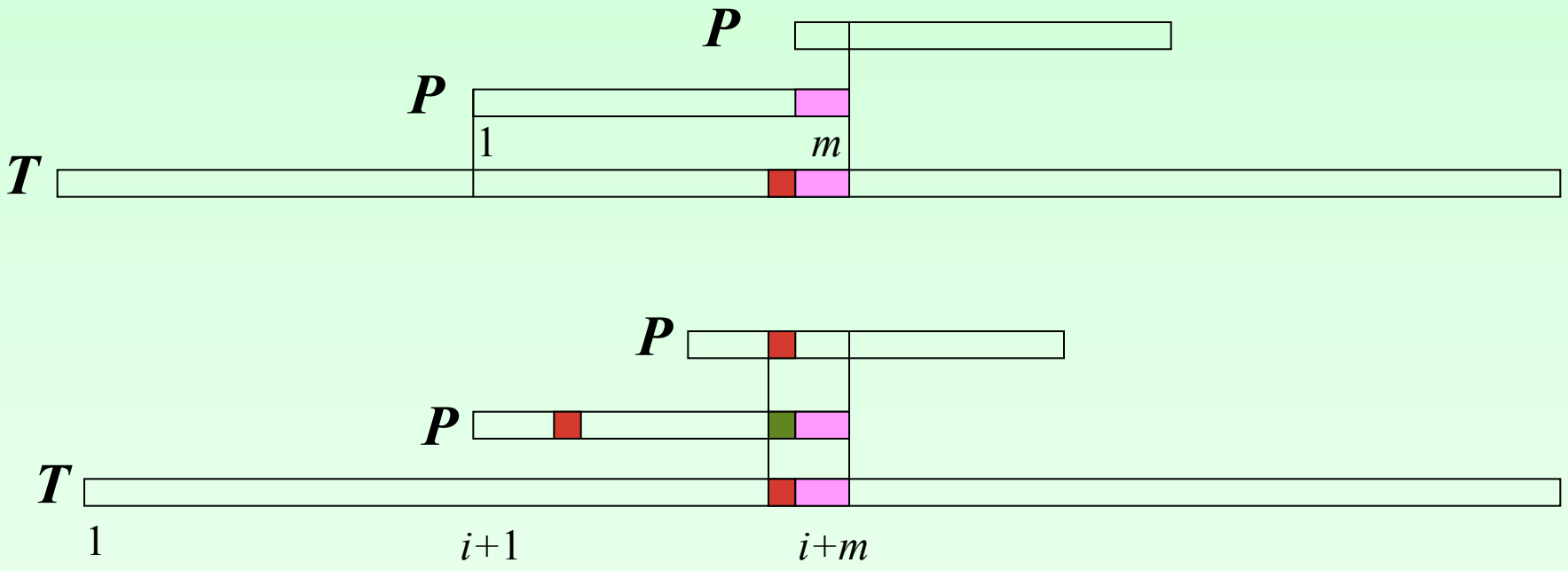
1. Правило «плохого символа».





Алгоритм Бойера-Мура

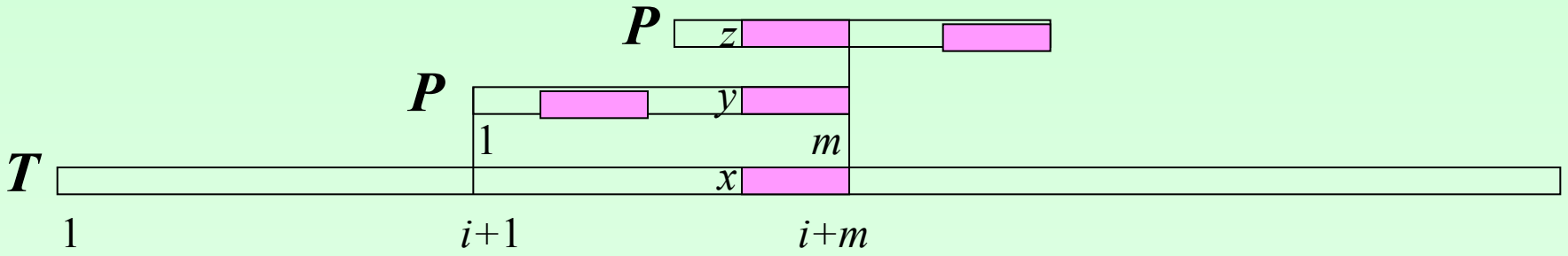
1. Правило «плохого символа».



$$\delta_1(a) = \begin{cases} m - j, & \text{где} \quad j = \max\{i \mid p_i = a\}, \\ m, & \text{если} \quad a \notin \{p_1, \ldots p_m\}. \end{cases} \qquad a \in \Sigma$$

Алгоритм Бойера-Мура

2. Правило «хорошего суффикса».



$$\delta_2(j) = j + 1 - rpr(j), \text{ где } 1 \le j \le m$$

$$rpr(j) = \max\{k \mid (P[j+1:m] = P[k:k+m-j-1])$$

$$and \quad ((k \le 1)\ or\ (p_{k-1} \ne p_j)\}$$

Поиск образцов. Алгоритм Shift-And

$$R[i, j] = \begin{cases} 1, & \text{если } P[1:i] = T[j-i+1:j], \\ 0, & \text{в остальных случаях.} \end{cases}$$

$R[m, j] = 1$: P в $(j - m + 1)$-й позиции T.

Пример. Пусть $\Sigma = \{a,b,c\}$, p=aabac, T=aabaacaabacab.

$R[3,3] = 1$, $R[4,4] = 1$, $R[5,5] = 0$;
$R[1,6] = 0$, $R[2,7] = 0$, $R[3,8] = 0$ …
$R[2,5] = 1$, $R[3,6] = 0$; $R[4,7] = 0$;
$R[1,7] = 1$, $R[2,8] = 1$, $R[3,9] = 1$, $R[4,9] = 1$, $R[5,11] = 1$

Алгоритм Shift-And

$$R[i+1, j+1] = \begin{cases} 1, & \text{если } R[i, j] = 1 \text{ и } p_{i+1} = t_{j+1}, \\ 0, & \text{в остальных случаях.} \end{cases}$$

Переход от j-го столбца R к (j+1)-му:
правый сдвиг R[*, j]
и ***And***-операция с S[*, i + 1], где $s_{i+1} = t_{j+1}$.

Пример. Пусть Σ = {a,b,c}, p=aabac, T=aabaacaabacab.

	a	b	c
0	1	1	1
1	1	0	0
2	1	0	0
3	0	1	0
4	1	0	0
5	0	0	1

R

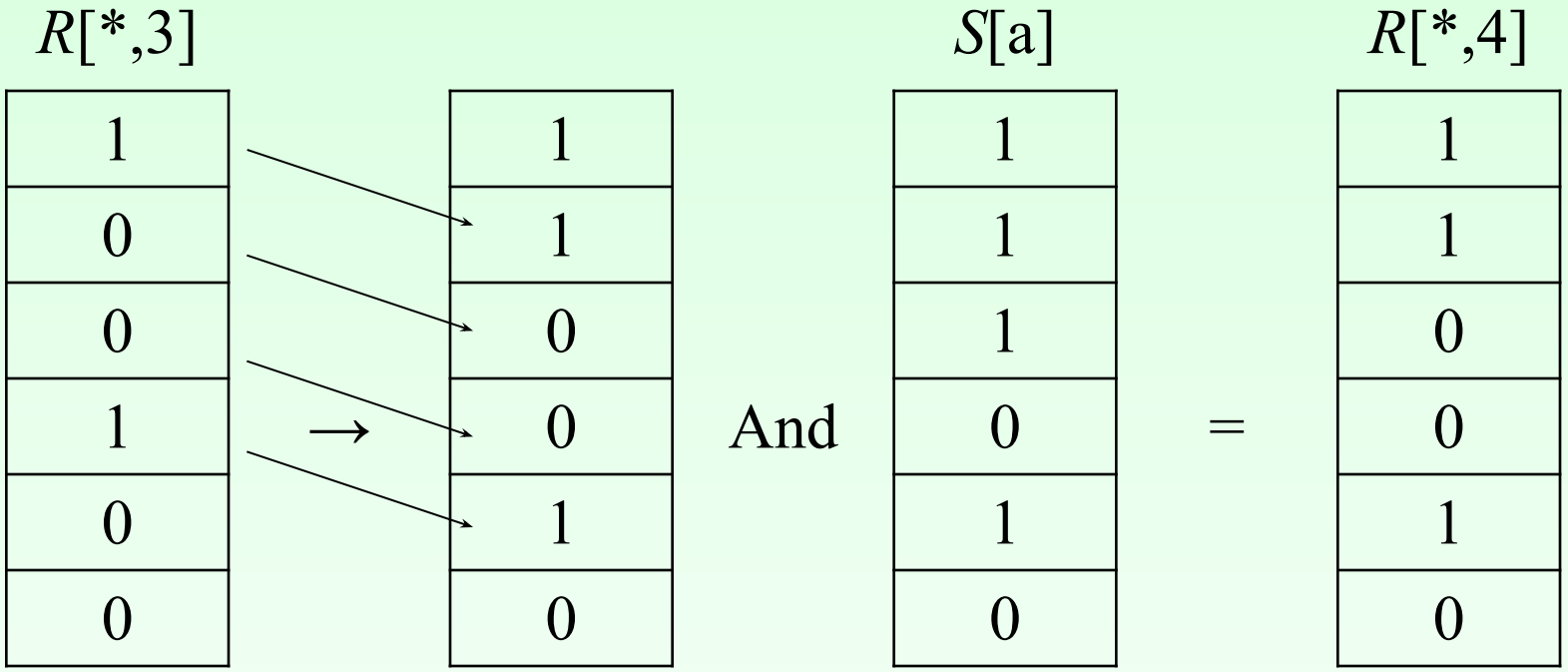
		a	a	b	a	a	c	a	a	b	a	c	a	b
	1	1	1	1	1	1	1	1	1	1	1	1	1	1
a	0	1	1	0	1	1	0	1	1	0	1	0	1	0
a	0	0	1	0	0	1	0	0	1	0	0	0	0	0
b	0	0	0	1	0	0	0	0	0	1	0	0	0	0
a	0	0	0	0	1	0	0	0	0	0	1	0	0	0
c	0	0	0	0	0	0	0	0	0	0	0	**1**	0	0

Алгоритм Shift-And

$$R[i+1, j+1] = \begin{cases} 1, & \text{если } R[i,j] = 1 \text{ и } p_{i+1} = t_{j+1}, \\ 0, & \text{в остальных случаях.} \end{cases}$$

Пример. Пусть Σ = {a,b,c}, p=aabac, T=aabaacaabacab.

Схема перехода от 3-го столбца *R* к 4-му:



Алгоритм Карпа-Рабина

***ns* : Σ → [0.. |Σ| - 1] -** порядок символов в **Σ.**

Пусть $s = |\Sigma|$. Тогда

$H(P) = ns(p_1) \times s^{m-1} + ns(p_2) \times s^{m-2} \ldots ns(p_{m-1}) \times s + ns(p_m)$ и

$H(T[i : i + m - 1]) = ns(t_i) \times s^{m-1} + ns(t_{i+1}) \times s^{m-2} \ldots ns(t_{i+m-2}) \times s + ns(t_{i+m-1})$.

Если $H(P) = H(T[i : i + m - 1])$ **-** образец встретился в *i*-й поз. текста.

Рекуррентное хеширование:

$H(T[i + 1 : i + m) = (H(T[i : i + m - 1]) - ns(t_i) \times s^{m-1}) \times s + ns(t_{i+m})$.

Схема Горнера вычисления H:

$H(P) = (\ldots(((ns(p_1) \times s + ns(p_2)) \times s + ns(p_3)) \times s + \ldots + ns(p_{m-1})) \times s + ns(p_m)$.

Пример. Σ = {acgt}, *P* = acat, *T* = ggacataccagac;

$H(P) = 0 \times 4^3 + 1 \times 4^2 + 0 \times 4^1 + 3 = 19;$

$H(T[1:4]) = 2 \times 4^3 + 2 \times 4^2 + 0 \times 4^1 + 1 = 161;$

$H(T[2:5]) = 2 \times 4^3 + 0 \times 4^2 + 1 \times 4^1 + 0 = 132 = (161 - 2 \times 4^3) \times 4 + 0;$

$H(T[3:6]) = 0 \times 4^3 + 1 \times 4^2 + 0 \times 4^1 + 3 = 19 = (132 - 2 \times 4^3) \times 4 + 3;$

Обобщения задачи поиска образца:

- Образцы с джокерами: *x* – любой символ
  Пример. *P = abxxcx* содержится в тексте gabvccbababcad дважды.

- Образцы, позиции которых заданы множествами символов A- [AG]-C-[CG]-[ACG]-[CT]-A
  A- [AG]-C-[CG]-¬T-*x*-A
  (AGCCAAA, AACCGCA…)
- Поиск образца с допустимым уровнем искажений:
  ACGTAC – ACTTAC – ACGTCC – ACTGTAC – ACTAC
- Поиск множества образцов
- Комбинации задач (например, поиск множества образцов, позиции которых заданы множествами символов)
- Образцы с переменными
  P = abXXcX : abttct; ababbabbcabb
- Параметризованные образцы: 2 алфавита: Σ и Π:
  Образцы abcXbbYYccZ и abcZbbXXccY π-согласованы

Алгоритм Ахо-Корасик

Задача. Задано множество образцов $\boldsymbol{P} = \{P_1, P_2, \ldots P_z\}$. Требуется **обнаружить все вхождения в текст *T* любого образца из *P*.**

i-й образец $P_i = p_{i1}p_{i2}\ldots p_{i,mi}$ имеет длину m_i; $p_{i,j} \in \Sigma$.

Текст $\boldsymbol{T} = t_1\ t_2\ \ldots\ t_N$, $t_k \in \Sigma$, $1 \le k \le N$.

Это обобщение называют множественной задачей точного поиска или задачей поиска по групповому запросу

Наивный алгоритм решает эту задачу путем поиска каждого образца из набора с использованием любого из рассмотренных выше линейных алгоритмов. Такой поиск имеет трудоемкость $O(zN + \sum_i m_i)$.

Эффективный алгоритм решения этой задачи имеет трудоемкость $O(N + \sum_i m_i)$.

Алгоритм Ахо-Корасик

- Этап предобработки: построение ДКА по исходному множеству образцов
- Этап поиска: однократный "прогон" текста через этот автомат.

1. Этап предобработки.

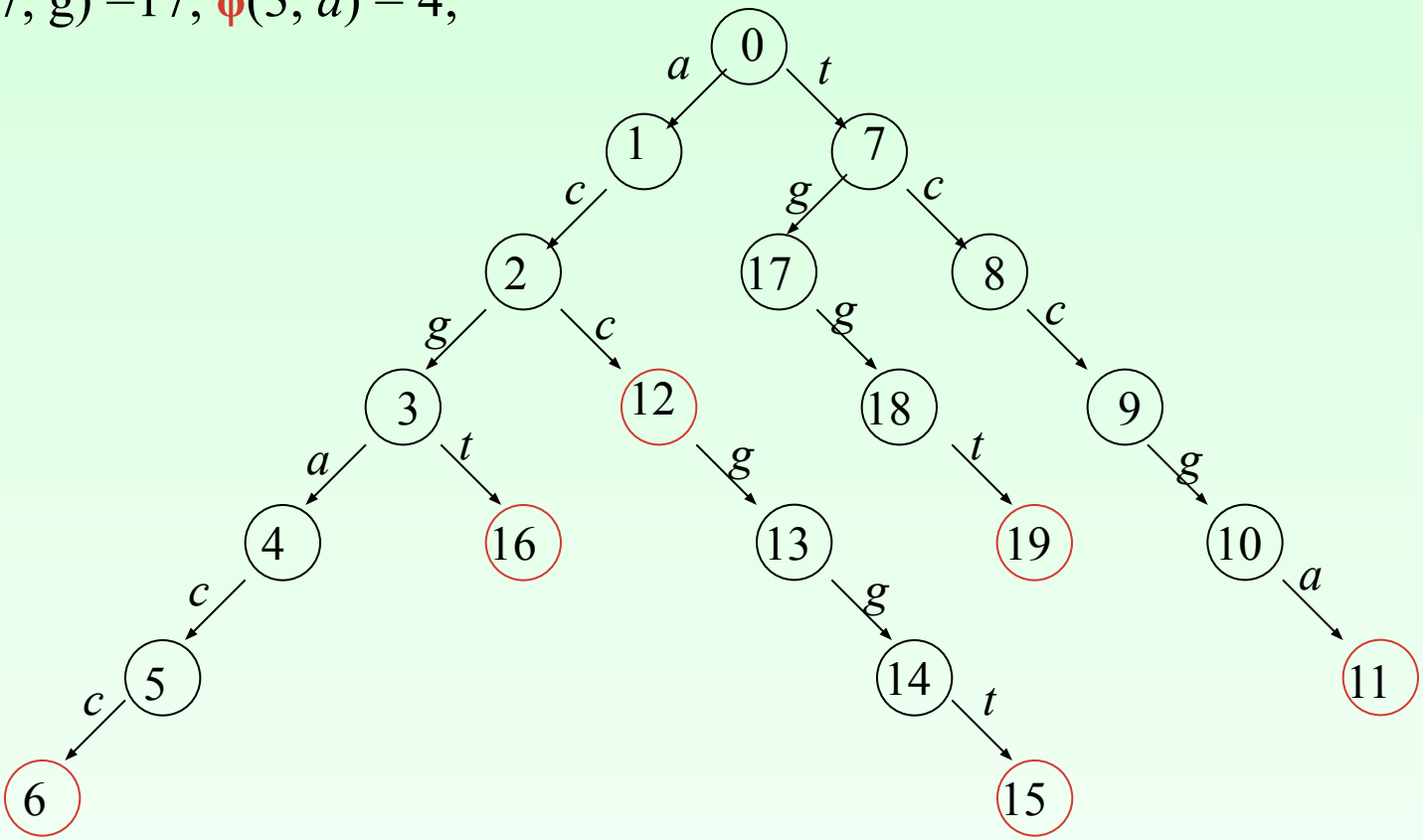
   Сначала строится "машина идентификации цепочек" ***Mp***.
   Работа машины ***Mp*** описывается тремя функциями:
   функцией переходов $\varphi(s,a)$ (s – состояние машины, $a \in \Sigma$),
   функцией отказов $f(s)$
   и функцией выходов $o(s)$.

Алгоритм Ахо-Корасик

Функция переходов $\varphi(s,a)=s'$, если существует выходящее из s ребро, помеченное символом "a" и связывающее состояния s и s'; в противном случае φ(s,a) = "fail" (ситуация, обозначаемая термином "отказ").

Пример. Пусть Σ = {a,c,g,t}; P = {acgacc, tccga, accggt, acgt, acc, tggt};

$\varphi(7, g) = 17$; $\varphi(3, a) = 4$;



Алгоритм Ахо-Корасик.

Построение $f(s)$: пусть $\varphi(s\_pred,a) = s$, $f(s\_pred) = s''$.

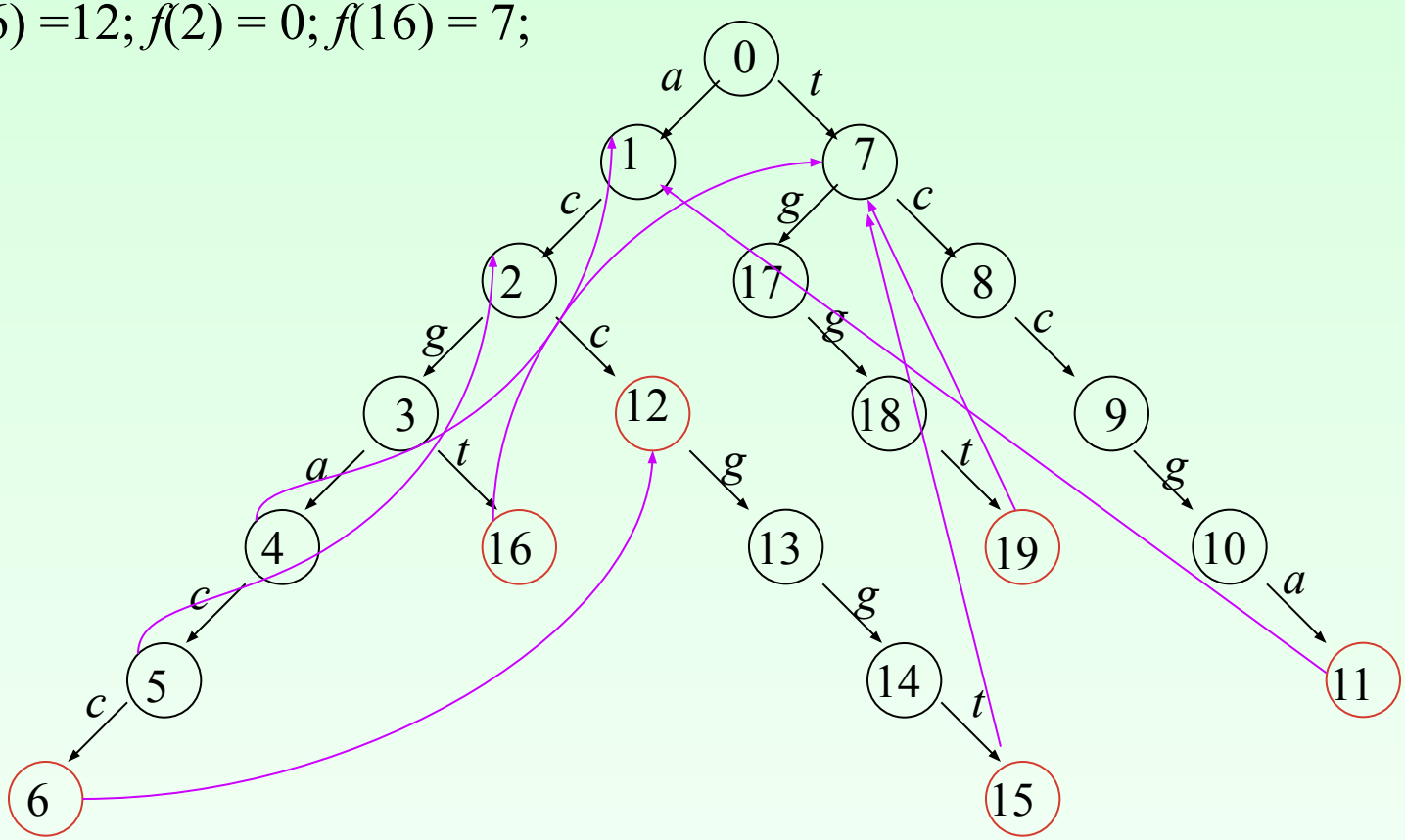
Metka : Если φ(s",a)<> fail, то f(s)= φ(s",a); o(s) := o(s) ∪ o(f(s)) ,

иначе s" := f(s"); goto Metka.

Порядок построения: по уровням дерева (структура «очередь»).

Пример. Пусть Σ = {a,c,g,t}; P = {acgatc, tccga, accggt, acgt, acc, tggt}; o(6)={1,4};

$f(6) = 12$; $f(2) = 0$; $f(16) = 7$;



Алгоритм Ахо-Корасик.

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
A	1	1	1	4	1	1	1	1	1	1	11	1	…							1
C	0	2	0	0	5	6	0	8	9	0	0	2								8
G	0	0	3	0	0	3	13	17	0	10	0	0								17
T	7	7	7	16	7	7	7	7	7	7	7	7								7