ТВП

Теория вычислительных процессов

Лекция №2

(версия 1.0)

Курсовые работы

1. Система управления макетом ДЖД — 1-2 ч.
2. Теория и практика применения MPI и OpenMP — 1 ч.
3. Развертывание вычислительного кластера — 1 ч.
4. Проверка гипотезы Гольдбаха (с использованием кластера) — 2 ч.
5. Управление роботом на основе знаков дорожного движения — 1 ч.
6. Распознавание текста песен на английском языке — 1 ч.
7. Распознавание текста песен на русском языке — 1 ч.
8. Синтез речи для системы голосового оповещения — 1 ч.
9. Программное обеспечение сбора информации из социальных сетей — 1 ч.
10. Управление жестами трехмерным изображением — 1 ч.
11. Управление голосом трехмерным изображением — 1 ч.
12. Распознавание лиц на фотографиях (в видеопотоке) — 2 ч.

Требования к курсовой работе

- Изучить предметную область;
- Спроектировать архитектуру;
- Спроектировать алгоритм расчета;
- Распараллелить расчет;
- Реализовать.

- Объем отчета: 30 стр.

Основы теории схем программ

Схема программы

```
begin
ввод(x);
y:=a;
L: if p(x) then goto L1;
y:=g(x,y);
x:=h(x);
goto L;
L1: вывод (y);
end;
```

Основы теории схем программ

Операторная схема — это одна из разновидностей модели алгоритма (программы).

Содержательно оператор — это часть алгоритма, всегда выполняющая одни и те же вычисления над значениями некоторых из переменных программы — аргументов оператора.

Основы теории схем программ

Оператор – это часть алгоритма, всегда выполняющая одни и те же вычисления над значениями некоторых из переменных – аргументов оператора.

Пример 1. Отыскание минимума *m* функции *f* для целых значений аргумента от 1 до *n*.

Рассмотрим алгоритм решения поставленной задачи.

1. Выполнить начальные присваивания: $m = f(1)$, $i = 2$.
2. Если $i > n$, то перейти к шагу 7.
3. Выполнить присваивание $u = f(i)$.
4. Если $u \geq m$, то перейти к шагу 6.
5. Выполнить присваивание $m = u$.
6. Выполнить присваивание $i = i + 1$ и перейти к шагу 2.
7. Вывести на экран *m*.

Обозначим каждый из приведенных операторов (1 – 7) $S1$, $S2$, $S3$, $S4$, $S5$, $S6$, $S7$.

m := f(1); i:= 2; (S1)

A: if i>n then go to E; (S2)

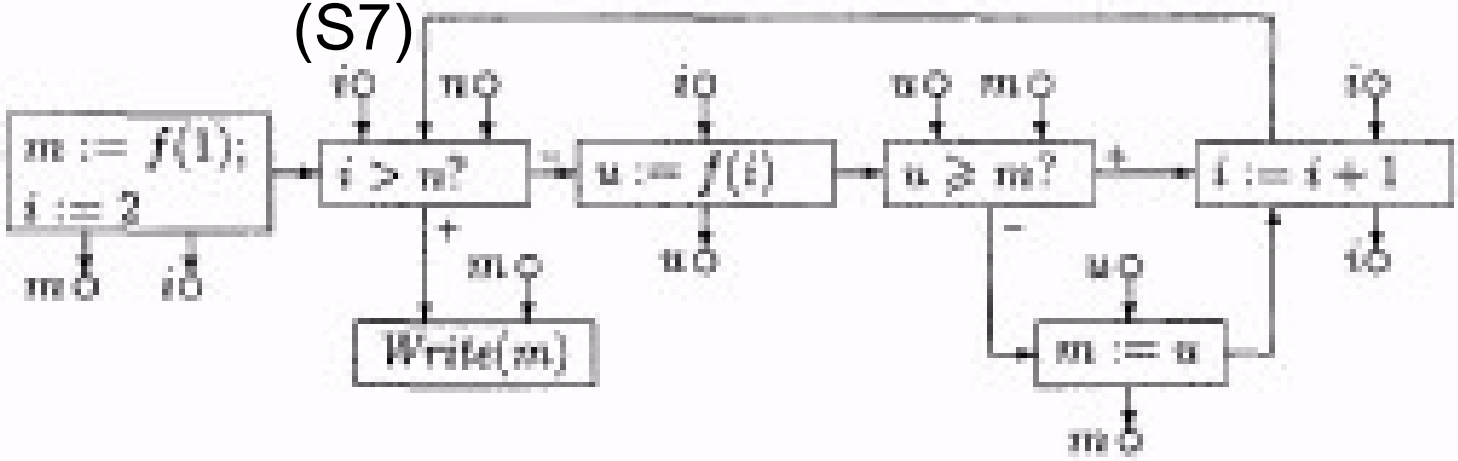
u := f(i); (S3)

if u ≥ m then go to B; (S4)

m:=u; (S5)

B: i := i + 1; go to A; (S6)

E: Write(m) (S7)



m := f(1); i:= 2; (S1)

A: if i>n then go to E; (S2)

u := f(i); (S3)

if u ≥ m then go to B; (S4)

m:=u; (S5)

B: i := i + 1; go to A; (S6)

E: Write(m) (S7)

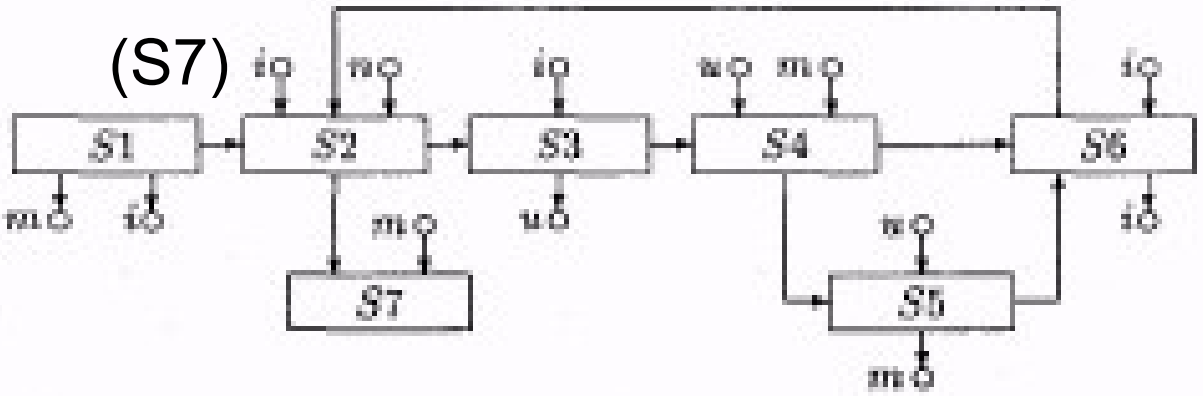
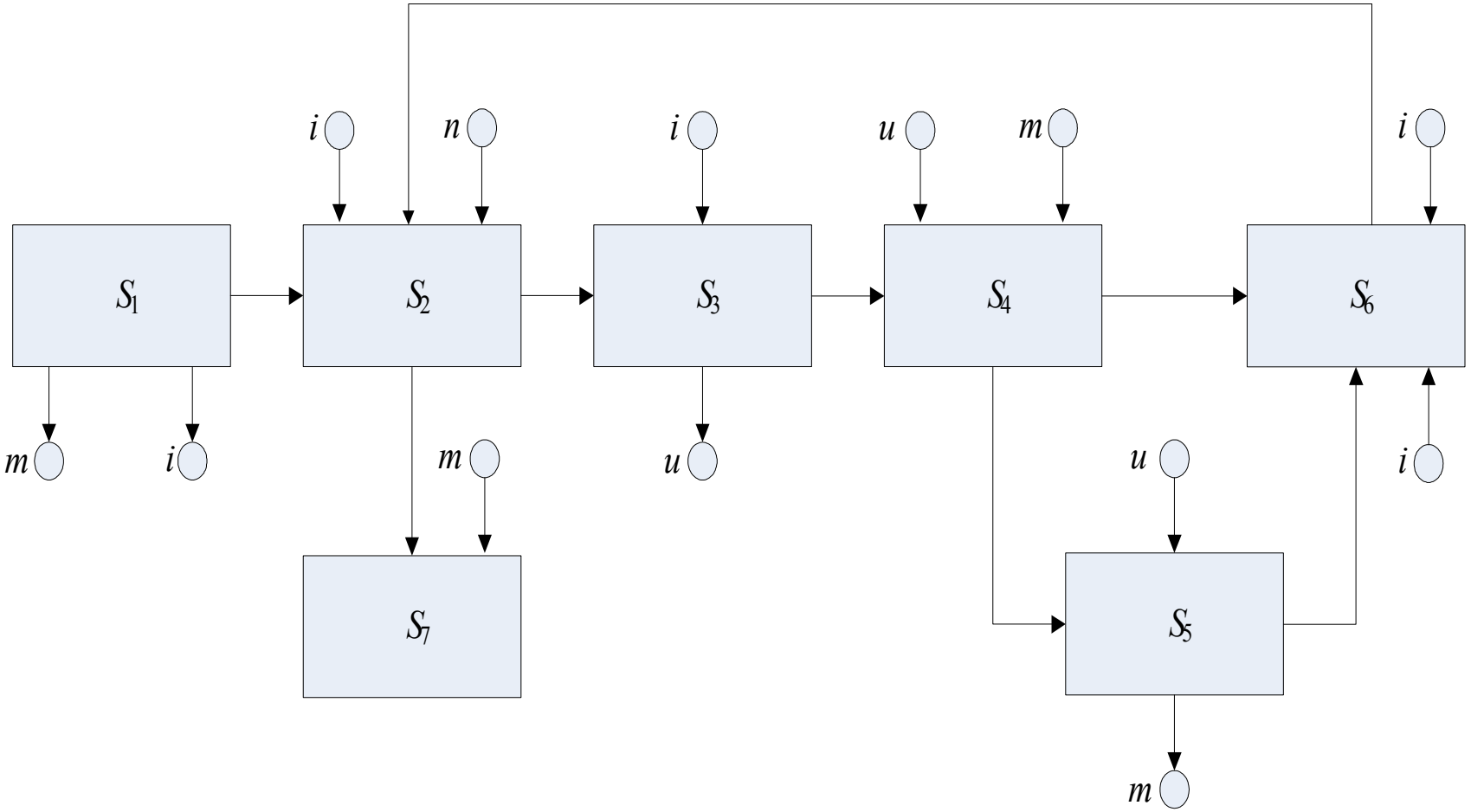


Рис. 2



Операторная схема алгоритма отыскания минимума целочисленной функции

Операторная схема

При представлении операторной схемы опираются на теоретико-множественное описание, согласно которому она задается пятеркой $<U, V, A, R, T>$, где

- $U = \{S_1, \ldots, S_m\}$ – множество операторов Sj,
- $V = \{x_1, \ldots, x_n\}$ – множество переменных xi,
- $A \subseteq V \times U$ – отношение «быть аргументом» ($x_i \, A \, S_j$),
- $R \subseteq U \times V$ – отношение «быть результатом» ($S_j \, R \, x_i$),
- $T \subseteq U \times U$ – отношение между оператором-предшественником и оператором-преемником ($S_j \, T \, S_k$), т.е. множество возможных переходов.

Операторная схема

Входом операторной схемы назовем оператор S_{in} со свойствами:1) S_{in} не имеет аргументов,2) S_{in} не имеет предшественников.

Выход операторной схемы – это оператор S_{out} со свойствами:1) S_{in} не имеет результатов,

2) S_{in} не имеет преемников.

Схема объединяет в себе ориентированные графы всех трех отношений A, R, T. Граф $<U \cup V, A \cup R>$ – двудольный. Тип (U или V) начала и конца каждой дуги определяет, какому из отношений соответствует эта дуга. Граф $< U, T >$ будем называть графом переходов.

Операторная схема

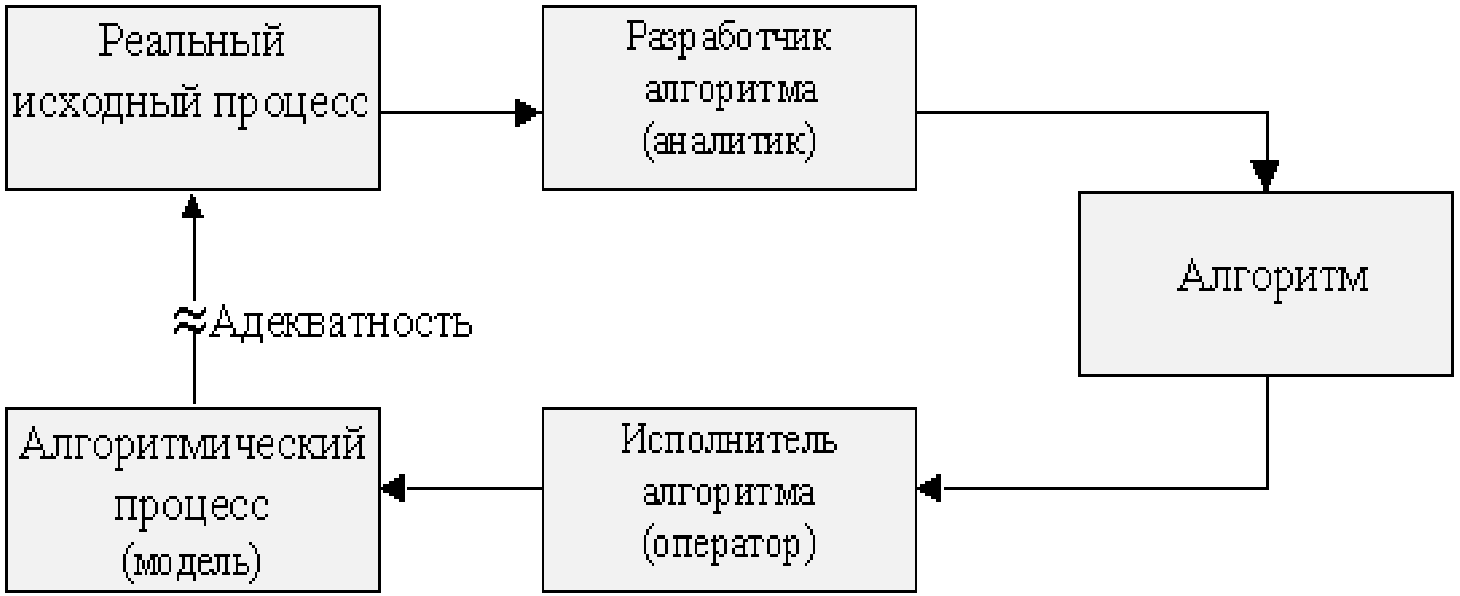
Граф переходов должен удовлетворять следующему условию связности: любая вершина (оператор) должна быть достижима из входа.

В теории операторных схем рассматриваются следующие основные проблемы:

- оценка трудоемкости алгоритма;
- методы исследования свойств алгоритма (например, завершаемость и связь между аргументами и результатами схемы в целом);
- экономия памяти (преобразования схемы, связанные с переобозначением переменных);
- эквивалентность операторных схем при преобразованиях, меняющих порядок исполнения операторов.

Алгоритм

Алгоритм есть точное предписание, которое задает вычислительный процесс нахождения значений вычислимой функции по заданным значениям ее аргументов.



Алгоритм

Свойство алгоритма (программы) – это некоторое утверждение о состоянии его переменных, т.е. о совокупности значений переменных из множества *V*. Утверждение о состоянии – это предикат, определенный на множестве состояний *W*, т.е. отображение множества *W* в множество *BV* = {*true*, *false*} логических значений. Состояния будем обозначать буквами *w*, …, предикаты – буквами p, q, … (возможно, со штрихами и индексами). Таким образом, если *w* – это состояние ($w \in W$), то $p(w)$ – логическое значение ($p(w) \in BV$).

Алгоритм

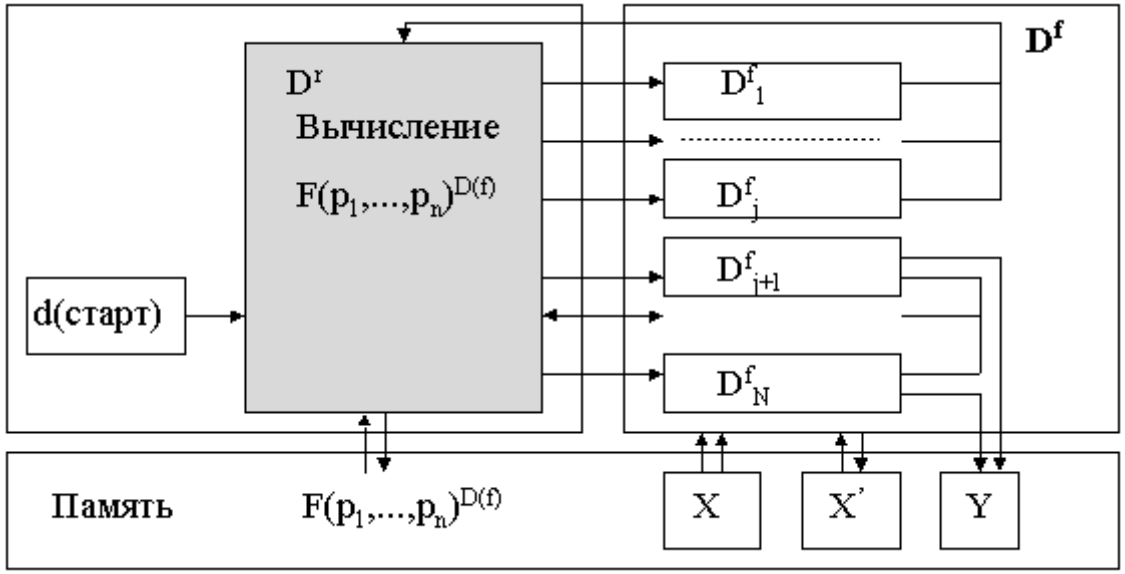
Операторы преобразуют некоторым способом состояния. Состояние, в котором исполняется оператор, определяет дугу графа *T*, по которой происходит переход к следующему исполняемому оператору. Таким образом, о состояниях и их свойствах, т.е. о предикатах, которым удовлетворяют состояния, целесообразно говорить не во время исполнения операторов, а в промежутках между исполнением оператора-предшественника и оператора-преемника. Поэтому предикаты естественно сопоставлять дугам графа переходов – размечать их этими предикатами. Пусть дуге a_i сопоставлен предикат *pi*. Совокупность всех предикатов p_i назовем предикатной разметкой дуг схемы.

Алгоритм

Любой алгоритмический процесс характеризуется следующими свойствами:

- результативностью;
- массовостью;
- дискретностью;
- детерминированностью;
- самоуправляемостью;
- альтернативностью и эквивалентностью;
- сложностью;
- адекватностью;
- ресурсоемкостью.

Математические модели алгоритмических процессов



Пример синтеза АП

$ay^2+by+c=0$

p_1	p_2	p_3	p_4	$F(p_1, p_2, p_3, p_4)^{D(f)}$
–	–	–	(данные **x** не введены)	D^f_1 - ввод исходных данных **x**= (a, b, c)
–	–	–	(данные **x** введены)	D^f_2 - вычисление $d = b^2 - 4ac$
($a \neq 0$)	**($d > 0$)**	–	(данные **x** введены)	D^f_3 - вычисление $y_1 = (-b + \sqrt{d}) / 2a$, $y_2 = (-b - \sqrt{d}) / 2a$, выдать y_1, y_2, остановить АП.
($a \neq 0$)	**($d = 0$)**	–	(данные **x** введены)	D^f_4 - вычисление $y = -b / 2a$, выдать y, остановить АП.
($a = 0$)	–	**($b \neq 0$)**	(данные **x** введены)	D^f_5 - вычисление $y = -c / b$, выдать y, остановить АП.
($a \neq 0$)	**($d < 0$)**	–	(данные **x** введены)	D^f_6 - выдать сообщение: “нет действительных корней”, остановить АП.

Пример синтеза АП

p^2_1	p^3_2	p^2_3	p^2_4	$F(P)^6$
–	–	–	0	1
–	–	–	1	2
1	1	–	1	3
1	0	–	1	4
0	–	1	1	5
1	2	–	1	6

p_1	p_2	p_3	p_4	p_5	$F(P)^6$
–	–	–	0	–	1
–	–	–	1	0	2
1	1	–	1	1	3
1	0	–	1	1	4
0	–	1	1	1	5
1	2	–	1	1	6

Пример синтеза АП

p_1	p_2	p_3	p_4	p_5	$F(P)^6$
–	–	–	0	–	1
–	–	–	1	0	2
1	1	–	1	1	3
1	0	–	1	1	4
0	–	1	1	1	5
1	2	–	1	1	6

p_1	p_2	p_3	p_4	p_5	$F(P)^6$
–	–	–	0	–	1
–	–	–	1	0	2
1	1	–	1	1	3
1	0	–	1	1	4
0	–	1	1	1	5
1	2	–	1	1	6
0	–	0	1	1	6

Схема отображения множества действий

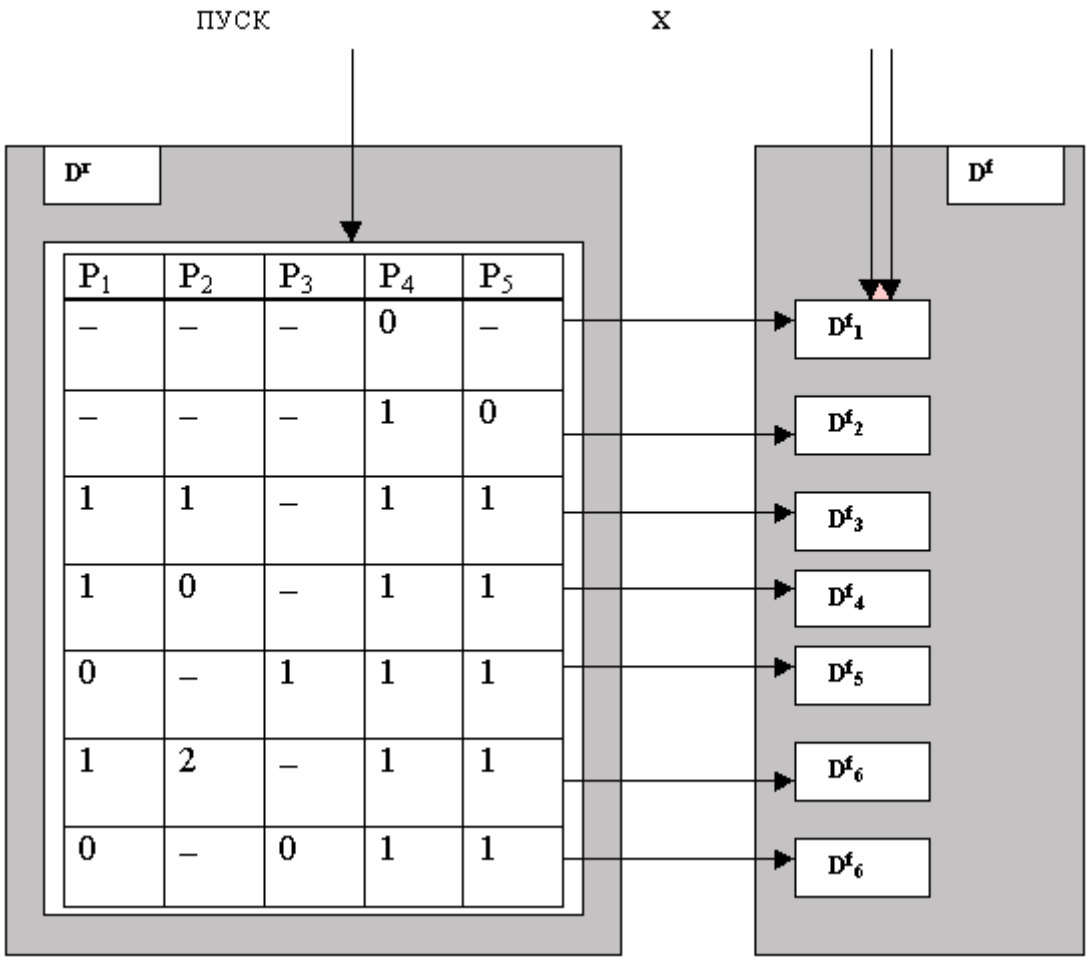
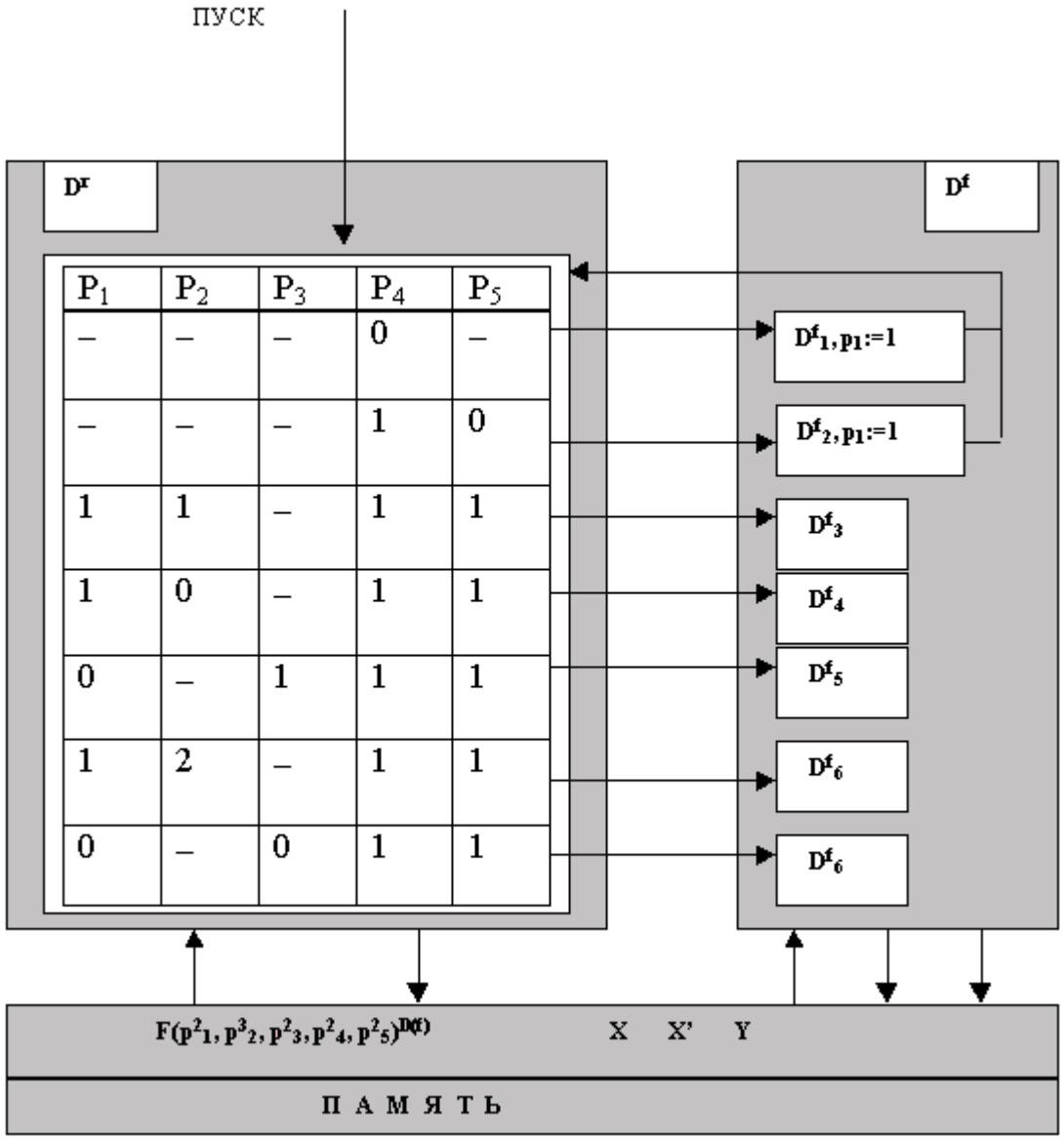


Схема отображения множеств $D^r \rightarrow D^f \rightarrow D^r$



Логическая схема алгоритма