АССОЦИАТИВНЫЕ ПАРАЛЛЕЛЬНЫЕ ВЫЧИСЛЕНИЯ

А.Ш. Непомнящая
ИВМ и МГ СО РАН
E-mail: anep@ssd.sscc.ru

Ключевые слова:

Контекстно-адресуемая (ассоциативная) память, вертикальная обработка информации, ассоциативный параллельный процессор (АПП), система вертикальной обработки (VPS)

План доклада:

- Основные книги по АПП.
- Основные достоинства и основные применения АПП.
- Модель класса. Пример вертикальной обработки информации.
- Модель систем вертикальной обработки (STAR-машина). Примеры операторов и базовых процедур языка STAR.
- Примеры классических алгоритмов на графах, для которых построены ассоциативные версии.

Терминология:

Array processor, associative array processor, massively parallel processor, associative parallel processor.

Основные книги по АПП:

- **C.Fernstrom, J.Kruzela, B.Svensson**. LUCAS Associative Array Processor. (LNCS, vol. 216, 1986).

- **Y.Fet.** Parallel Processing in Cellular Arrays. (Tauton, UK, Research Studies Press, 1995).

- **Foster C. C**. Content Addressable Parallel Processors. (Van Nostrand Reinhold, 1976).

- **A. Krikelis, C. Weems**. Associative Processing and Processors. (IEEE Computer Society Press, 1997).

- **J.L.Potter.** Associative Computing: A Programming Paradigm for Massively Parallel Computers. (Plenum Press, New York and London, 1992).

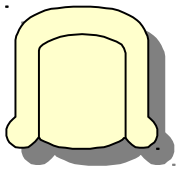
Основные достоинства АПП

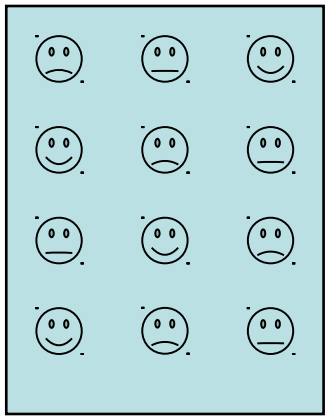
- Параллелизм по данным на базовом уровне.
- Массовый параллельный поиск по содержимому памяти.
- Обработка неупорядоченных данных.
- Использование двумерных таблиц в качестве базовых структур данных.
- Базовые операции поиска и арифметические операции выполняются за время, пропорц. числу битовых столбцов в заданной матрице.

Основные применения АПП

- Обработка сейсмических данных.
- Обработка изображений.
- Реляционные базы данных.
- Базы знаний.
- Экспертные системы.
- Алгоритмы на графах.

- Модель класса





Вертикальная обработка

1001	9	p = 3, s = 3
1100	12	p = 3, s = 9
1111	15	p = 1, s = 19
0100	4	p = 2, *s = 40*
	40	

s = p + 2s, где p – число единиц в текущем столбце.

◊ SIMD — **S**ingular **I**nstruction stream **M**ultiple **D**ata stream.

◊ *Системой вертикальной обработки* (VPS) назовем АПП типа SIMD с вертикальной обработкой и прост. ПЭ.

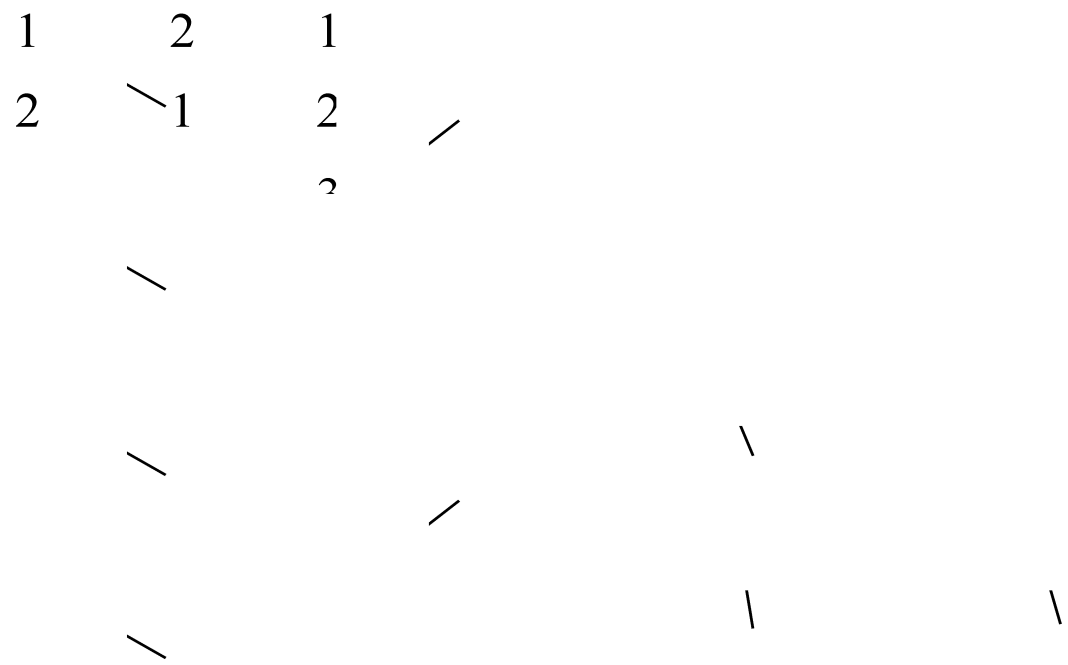
◊ SIMD $\supset$ VPS $\supset$ {Staran, Aspro, ES-27-20, DAP, MPP, CM-2}

◊ ES-27-20 — *отечеств*. АПП типа Staran.

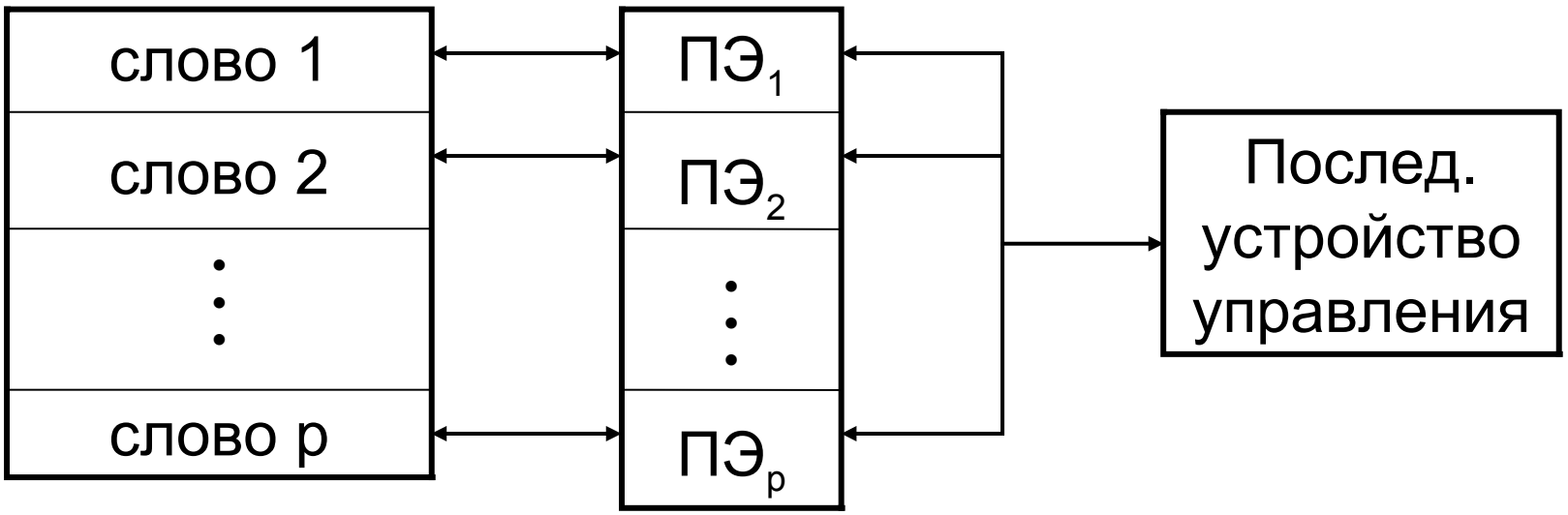
Характеристика систем вертикальной обработки

- CM-2 (1988): 65.536 PEs, hypercube (12 D) and NEWS, 64 kbits;
- DAP-610 (1976): 4096 (64×64) PEs, NEWS, 64 kbits;
- MPP (1979): 16.384 (128 × 128) PEs, NEWS, 64 kbits;
- STARAN (1974): up to 8.192 PEs, Flip, 1kbits.

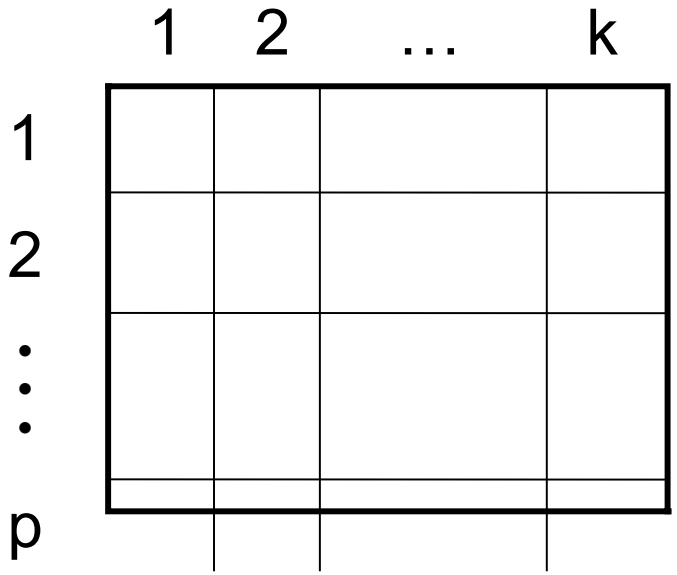
Соединение процессорных элементов в АПП STARAN



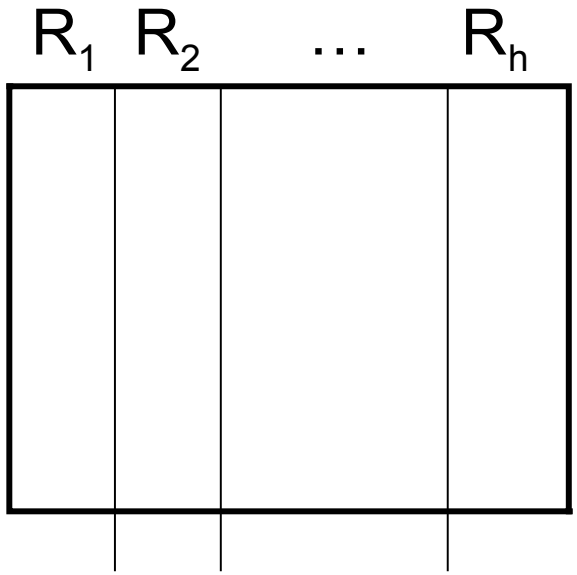
STAR-машина


слово 1
слово 2
слово p
ПЭ1
ПЭ2
ПЭp
Послед. устройство управления

Массив данных



Устройство ассоциативной обработки



Основные операции языка STAR

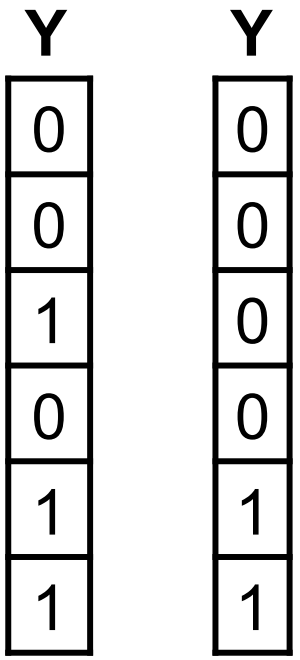
Операции для слайсов:

- SET(Y) записывает в слайс Y все единицы.
- CLR(Y) записывает в слайс Y все нули.
- Y(i) выделяет i - ю компоненту в слайсе Y.
- NUMB(Y) выдает число единиц в слайсе Y.
- FND(Y) выдает порядковый номер позиции первой единицы в слайсе Y.

- STEP(Y) выдает такой же результат, как FND(Y), затем сбрасывает первую (старшую) единицу.

- CONVERT(Y) возвращает строку, i - й бит которой совпадает с Y(i).

Побитовые логические операции:
X and Y, X or Y, not X, X xor Y.



Предикат SOME(Y) возвращает *true,* если в Y имеется хотя бы одна комп. '1'.

Операции для строк:

аналогичны операциям для слайсов. Для них имеются следующие предикаты: SOME(w), EQ(v,w), NOTEQ(v,w), LESS(v,w), LESSEQ(v,w), GREAT(v,w), GREATEQ(v,w).

Операции для матриц:

ROW(i,T) выделяет i - ю строку в матрице T.

COL(i,T) выделяет i - й столбец в матрице T.

SIZE(T) выдает число битовых столбцов в T.

Примеры операторов в языке STAR

Оператор присваивания:

k:=FND(Y); k:=STEP(Y); k:=NUMB(Y);

k:=Y(i); Y(i):=‘0’; k:=w(i);

Условный оператор:

if SOME(Y) then

if Y(i)=‘0’ then

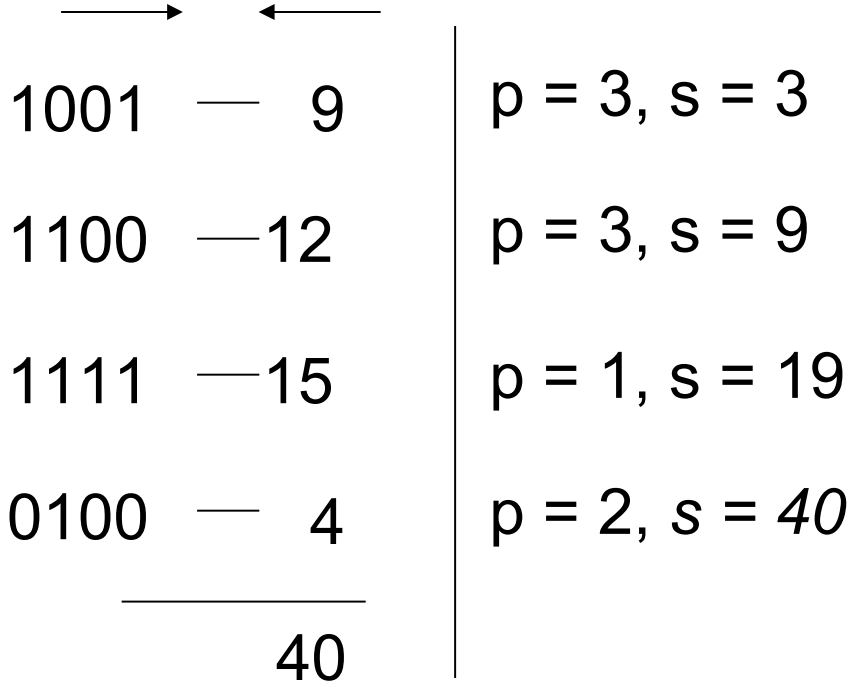
if w(i)=‘1’ then

Оператор итерации:

while SOME(Y) do

Ассоциативный алгоритм для нахождения суммы десятичных чисел

Вертикальная обработка



s = p + 2s, где p – число единиц в текущем столбце.

Примеры базовых процедур

```
procedure SUM(T: table; var s: integer);
      var X: slice; i,k,p: integer;
Begin s:=0; k:=SIZE(T);
    for i:=1 to k do
        begin X:=COL(i,T);
            p:=NUMB(X);
            s:=2s+p
        end;
End;
```

Проверка принадлежности v ∈ T

v

1	0	1	1

T

1	0	1	0
0	0	1	1
1	0	1	1
1	0	1	1
1	0	1	1
1	0	1	0

X

1
0
1
0
1
1

Z

0
0
1
0
1
0

Процедура MATCH

```
procedure MATCH(T: table; X: slice; v: word;
    var Z: slice);
var Y: slice; i,k: integer;
Begin Z:=X; k:=SIZE(T);
     for i:=1 to k do
         begin Y:=COL(i,T);
            if v(i)='1' then Z:=Z and Y
            else Z:=Z and (not Y)
         end;
End;
```

Ассоциативный алгоритм для нахождения минимального элемента

T			
1	0	0	0
0	0	1	1
1	0	1	1
1	0	1	1
1	0	0	1
1	0	1	0

X
1
0
1
0
1
1

Z
1
0
0
0
0
0

Процедура MIN

```
procedure MIN(T: table; X: slice; var Z: slice);
var i,k: integer; Y: slice;
Begin Z:=X; k:=SIZE(T);
  for i:=1 to k do
      begin Y:=COL(i,T); Y:=Z and (not Y);
          if SOME(Y) then Z:=Y
       end;
End;
```

Сортировка методом всплывания пузырька

```
procedure BUBBLE(T: table; X: slice;
    var P: table);
var i: integer; Y,Z: slice;
Begin CLEAR(P); Y:=X;  i:=0;
     while SOME(Y) do
        begin i:=i+1; MIN(T,Y,Z);
             COL(i,P):=Z;
             Y:=Y and (not Z);
        end;
End;
```

Библиотека стандартных процедур

◊ *Базовые процедуры для нечисловой обработки:*

Процедуры, применяемые к одной матрице. (MATCH, MIN, MAX, GEL, BUBBLE,…)

Процедуры, применяемые к двум матрицам. (HIT, SETMIN, SETMAX,…)

◊ *Базовые процедуры для числовой обработки:*

(ADDV, ADDC, SUBTV)

Примеры базовых процедур

HIT(T,F,X,Z) возвращает слайс Z, в котором Z(i) = '1', когда ROW(i,T) = ROW(i,F) и X(i) = '1'.

SETMIN(T,F,X,Z) возвращает слайс Z, в котором
Z(i) = '1', когда ROW(i,T) = ROW(i,F) и X(i) = '1'.

TCOPY1(T,j,h,F) записывает в F h столбцов из матрицы T, начиная с (1+(j-1)h)-го столбца.

ADDV(T,F,X,R) записывает в матрицу R результат одновременного сложения соответствующих строк матриц T и F, отмеченных '1' в слайсе X.

Определения

Пусть G = (V,E) - это *ориентированный взвешенный* граф, где V = {1,2,...,n } – множ. вершин, E – множ. дуг. Пусть wt(e) – это функция веса.

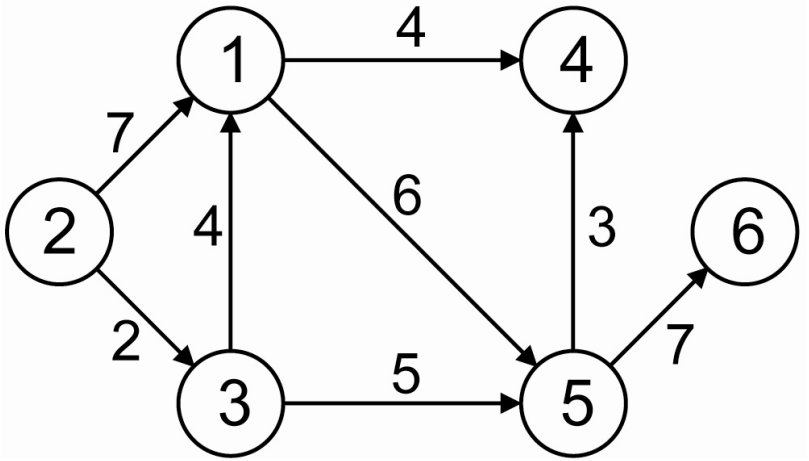
Послед. вершин $v_1, v_2, \ldots, v_k$ – это *путь* из v_1 в v_k, где $(v_i, v_{i+1}) \in E$ $(1 \le i \le k-1)$.

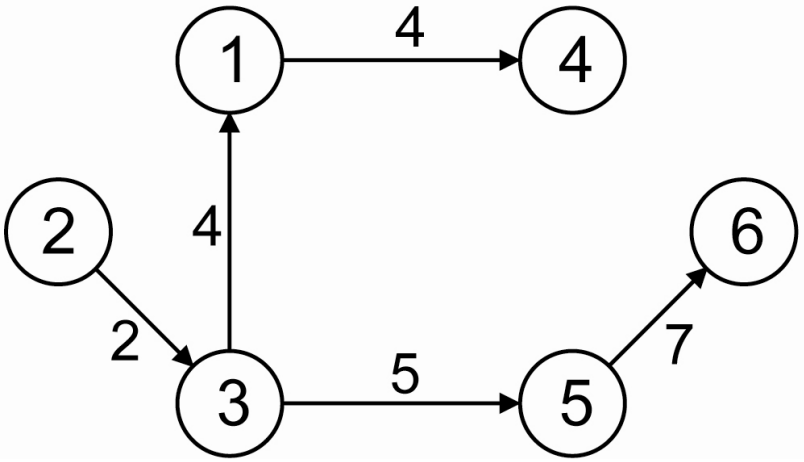
Кратчайший путь между двумя вершинами в G – это путь с *минимальной суммой* весов соотв. дуг.

Дерево кратчайших путей с корнем v_1 – это связный подграф *без циклов*, включающий все вершины графа G, и для $\forall\, v_j$ существует единств. путь из вершины v_1.

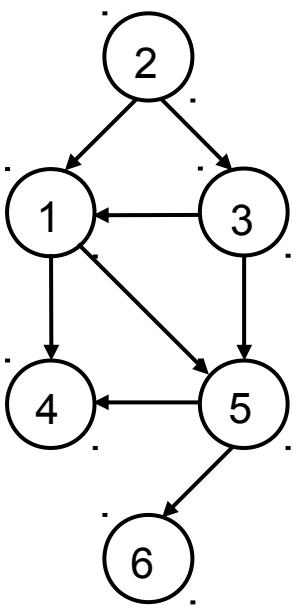
Транзитивным замыканием ориентир. графа G=(V.E) является ориентир. граф G*=(V,E*), в котором (u,v) $\in$ E* тогда и только тогда, когда верш. v достижима из верш. u.

Граф и дерево кратчайших путей





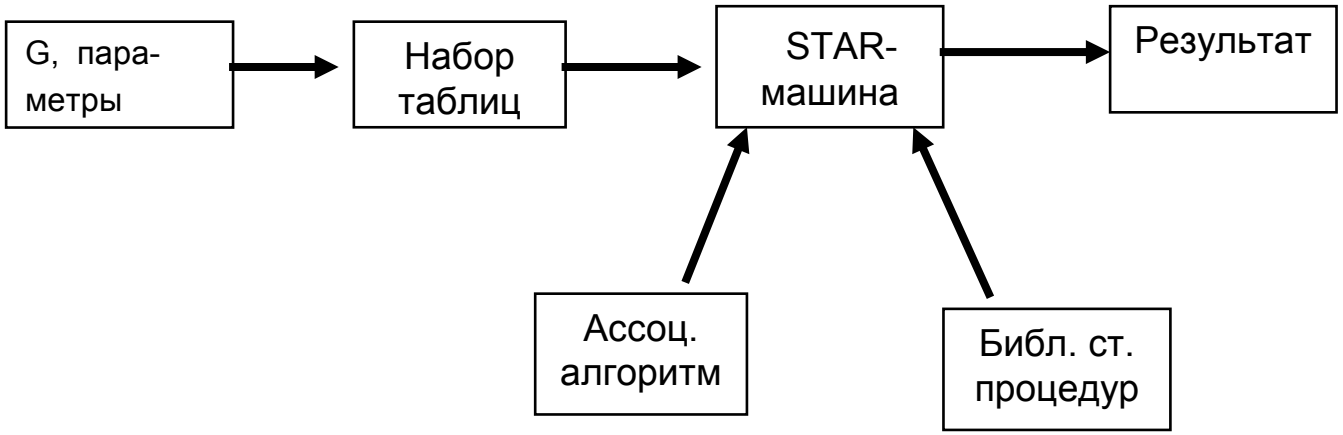
Два способа задания графов на STAR-машине



	Graph *G*		
	Left	Right	X
1	001	100	1
2	001	101	1
3	010	001	1
4	010	011	1
5	011	001	1
6	011	101	1
7	101	100	1
8	101	110	1

	Adj					
	1	2	3	4	5	6
1		1	1			
2						
3		1				
4	1				1	
5	1		1			
6					1	

Схема выполнения ассоциативных алгоритмов на STAR- машине



Алгоритмы на ориентированных графах, представленные на STAR-машине

- Нахождение транзитивного замыкания ориентированного графа (алгоритм Уоршалла).
- Алгоритмы Итальяно для динамической обработки транзитивного замыкания.
- Нахождение кратч. расстояний между любыми парами вершин (алгоритм Флойда).
- Нахождение кратч. расстояний и кратчайших путей от заданной верш. до ост. вершин графа (алгоритм Дейкстры и алгоритм Беллмана-Форда).
- Алгоритмы динамической обработки дерева кратчайших путей.
- Алгоритмы Рамалингама для динамической обработки подграфа кратчайших путей.

Задачи:

- Построить систему, моделирующую выполнение ассоциативных параллельных алгоритмов, записанных на языке STAR. (Систему можно реализовать либо на языке Borland Delphi 7, либо на языке C++). Построить группу тестов для проверки правильности этой системы.
- Построить систему визуализации выполнения ассоциативных параллельных алгоритмов, записанных на языке STAR.

Заключение

- Приведена общая характеристика АПП с вертикальной обработкой информации.
- Приведена STAR-машина, которая моделирует работу таких процессоров.
- Описан язык STAR. Приведены примеры стандартных процедур этого языка.
- Перечислены классические алгоритмы на *ориентированных* графах, которые были эффективно представлены на STAR-машине.