Организация метавычислений на распределенных и облачных вычислительных системах

Цель

Создание метасистемы переходов от последовательных вычислений к распределенным и облачным вычислениям

Задачи

1. Анализ методов построения и оптимизации реляционных схем и параллельных алгоритмов для распределенных вычислений и методов организации вычислений в облачных кластерах, инструментальных средств, предназначенных для формализованного представления алгоритмов.

2. Разработка метода оценки минимального количества необходимых процессоров для решения вычислительной задачи и определение совокупности метаданных, необходимых при принятии решения о модификации параллельного алгоритма с целью улучшения его эффективности.

Задачи

3. Разработка методов оптимизации параллельного алгоритма по времени выполнения и числу процессоров по отдельности и в совокупности.

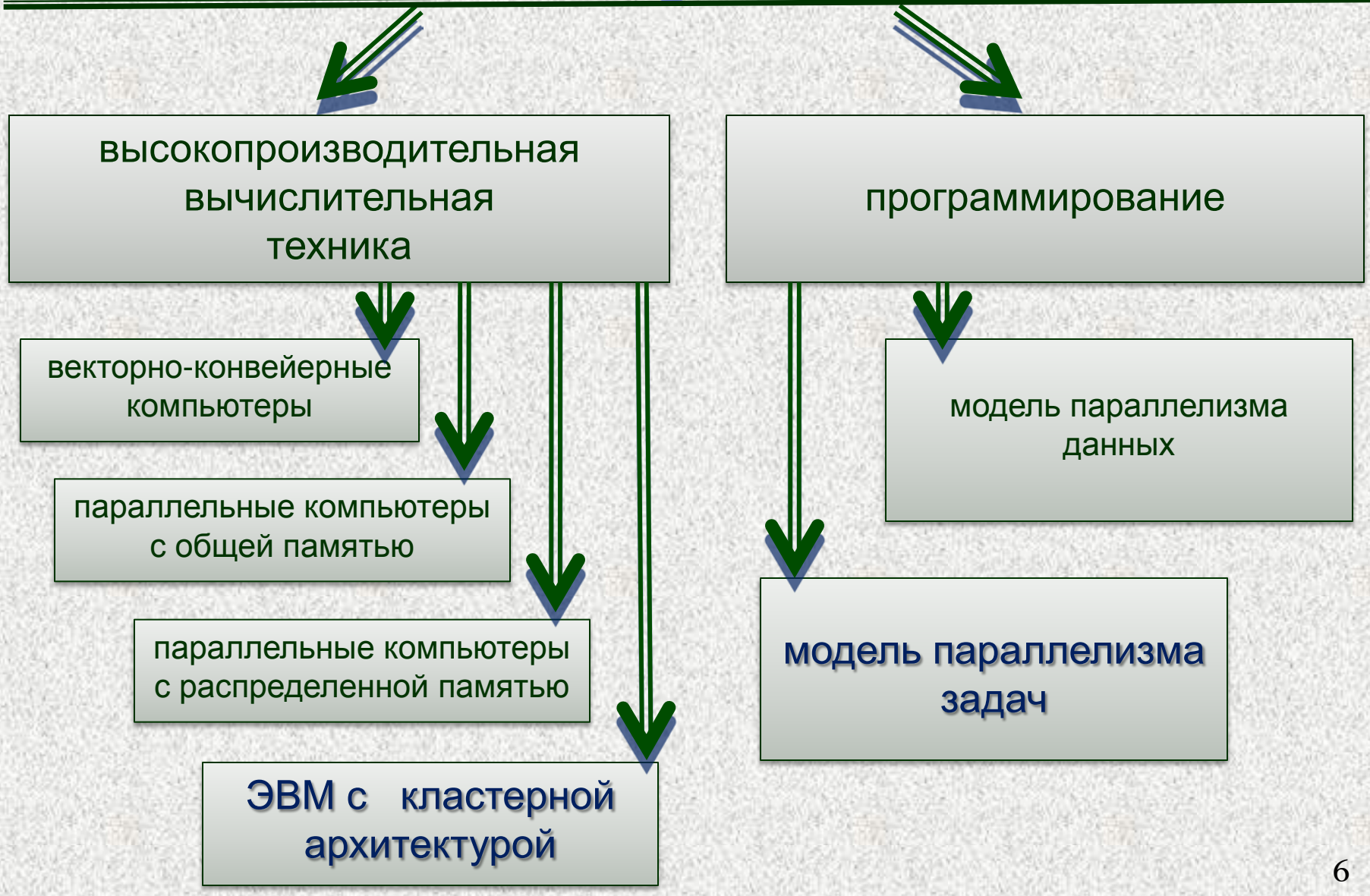
4. Разработка программного обеспечения для апробации полученных алгоритмов и методики их применения в зависимости от количественных характеристик параллельного алгоритма и параметров очередного шага оптимизации.

Задачи

5. Построение онтологической модели облачного кластера и на ее основе разработка методов распределения параллельных программ между ролями-сотрудниками при организации облачных вычислений.

6. Разработка матричного представления операций реляционной алгебры с помощью аппарата теории графов и численных методов создания эффективных реляционных схем, обладающих необходимым внутренним параллелизмом.

Увеличение скорости вычислений



Проектирование параллельных алгоритмов

Орграфы в качестве объектов вместе с функторами в качестве морфизмов образуют категорию, обозначаемую через OGR.

Переход от информационного графа к матрице или списку:

$$F : OGR \to M :$$

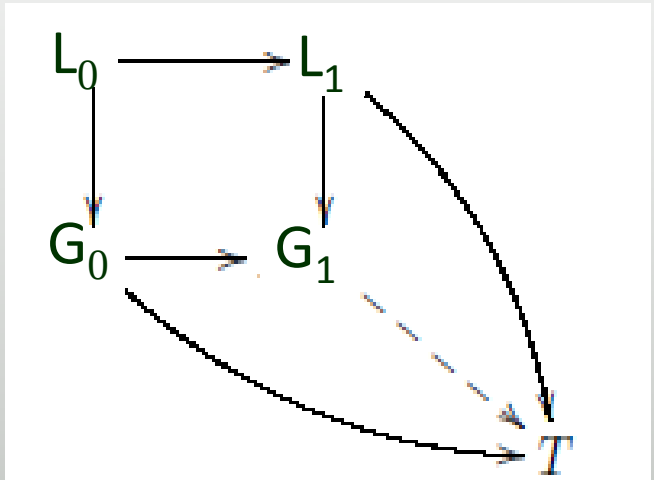
$$OGR \ni X \rightsquigarrow F(X) \in M \quad \text{и}$$

$$F(fg) = F(f)F(g)$$

где: OGR – категория орграфов,
M – прямоугольных разреженных матриц :

$$f \in Hom_{OGR}(X,Y),\ F(f) \in Hom_M\left(F(X),F(Y)\right)$$

pushout

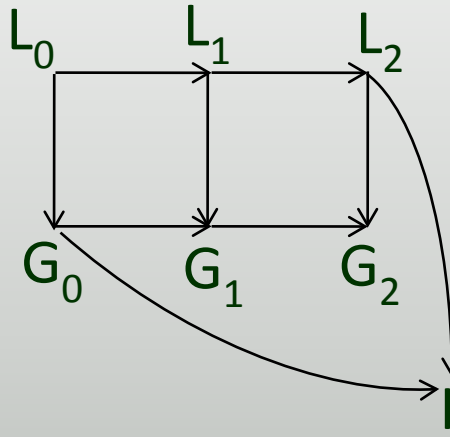


где: $G_0, G_1 \in$ OGR, $L_0, L_1 \in L$.

Оптимизация алгоритма по числу процессоров

1. Разбиение совокупности всех операций алгоритма на группы (построение параллельной формы)

2. Перемещение вершин между группами, оптимизация по числу процессоров



где: $G_0, G_1, G_2 \in OGR$, $L_0, L_1, L_2 \in L$.

Минимальная ширина информационного графа

1

Оценка минимальной ширины графа с учетом числа групп:

$$d' = \max_{1 \le k \le m} \left\lceil \frac{\sum_{i=k}^{m} d_i}{s - k + 1} \right\rceil$$

d_i - ширина i-й группы,
m – число групп,
S – высота графа,
k – номер группы

3

Разность между шириной графа и числом входных/выходных вершин

$$\Delta_{вх} = d' - n_{вх}$$

$$\Delta_{вых} = d' - n_{вых}$$

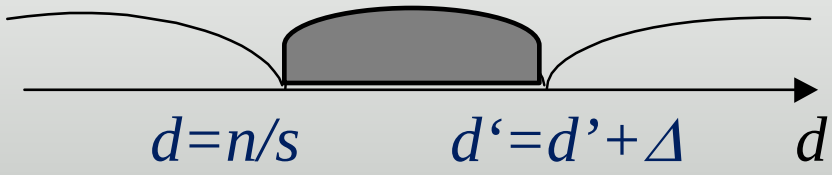
$n_{вх}$ – число входных вершин,
$N_{вых}$ – число выходных вершин

2

Приращение плотности групп в зависимости от $\Delta_{вх}$ и $\Delta_{вых}$

$$\Delta = \begin{cases} \dfrac{\Delta_{вх} + \Delta_{вых}}{g - 2}, (\Delta_{вх} \succ 0) \cap (\Delta_{вых} \succ 0) \\ 0, \textit{в остальных случаях} \end{cases}$$

4



Оптимизация алгоритма по числу процессоров

№	(V1,V2)	
1	1	2
2	1	5
3	2	15
4	3	4
5	4	5
6	4	10
7	5	0
8	6	10
9	6	7
10	7	14
11	8	6

№	(V1,V2)	
12	8	9
13	9	12
14	9	10
15	10	11
16	10	7
17	11	0
18	12	13
19	13	16
20	14	16
21	15	16
22	16	0

$M_1=\{1,3,8\}$
$M_2=\{2,4,6,9\}$
$M_3=\{15,5,10,12\}$
$M_4=\{7,11,13\}$
$M_5=\{14\}$
$M_6=\{16\}$

$M_1=\{1,3,8\}$
$M_2=\{2,4,6,9\}$
$M_3=\{5,10,12,15\}$
$M_4=\{7,11,13\}$
$M_5=\{14\}$
$M_6=\{16\}$

Список смежности – совокупность пар смежных вершин (v_i,v_j), где вершина v_i является начальной вершиной, а вершина v_j – конечной для ребра (v_i,v_j) информационного графа G.

1.

$V=V_1-V_2=\{1,3,8\} \Rightarrow M_1=\{1,3,8\}$

$V=V_1-V_2=\{2,4,6,9\} \Rightarrow M_2=\{2,4,6,9\}$

$V=V_1-V_2$

2.

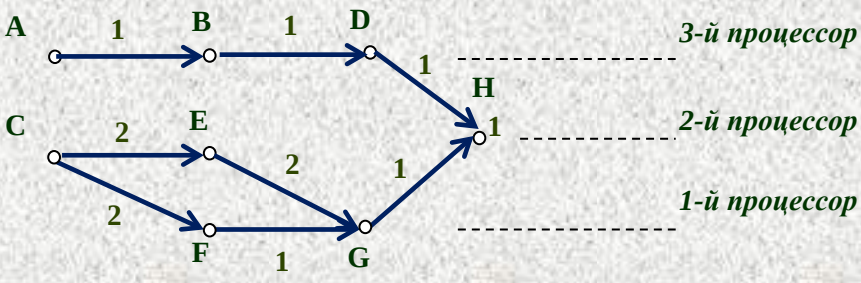
$M_5=\{14\}$, $V_{21}(M_4)=\{14\}$,
$V=M_5 \cap V_{21}(M_4)=\{14\} \neq \varnothing$
$V_{23}(M_4)=\{16\}$, $V_{22}(M_4)=\{0\}$
$V=M_5 \cap V_{22}(M_4)=\varnothing$
$V=M_5 \cap V_{23}(M_4)=\varnothing$

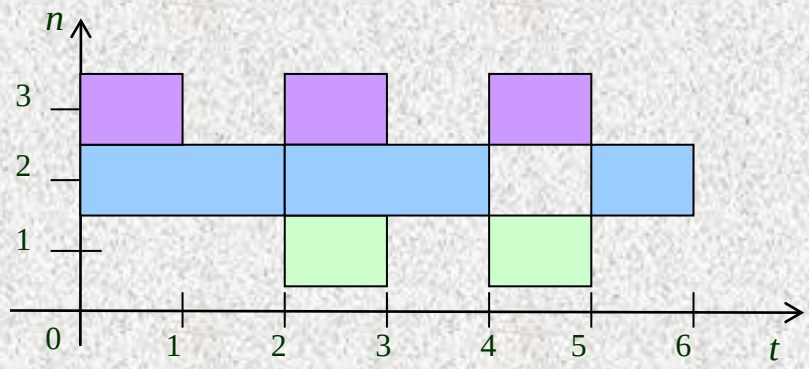
Если $V=M_i \cap V_{2j}(M_{i-1})=\varnothing$, то $M_{i-1,j} \rightarrow M_i$

Оптимизация алгоритма по числу процессоров

Матрица смежности	Списки смежности	Списки следования
Группы вершин		
$M_1=\{1,3,8\}$ $M_2=\{4,6,9\}$ $M_3=\{10,12,2\}$ $M_4=\{7,5,15\}$ $M_5=\{14,11,13\}$ $M_6=\{16\}$	$M_1=\{1,3,8\}$ $M_2=\{4,6,9\}$ $M_3=\{10,12,2\}$ $M_4=\{7,5,15\}$ $M_5=\{14,11,13\}$ $M_6=\{16\}$	$M_1=\{1,3,8\}$ $M_2=\{9,4,6\}$ $M_3=\{2,10,12\}$ $M_4=\{7,13,15\}$ $M_5=\{14,5,11\}$ $M_6=\{16\}$
Число проходов		
2	2	1
Трудоемкость		
$O(n^2)$	$O(n^2)$	$O(n^2)$

Оптимизация алгоритма по времени



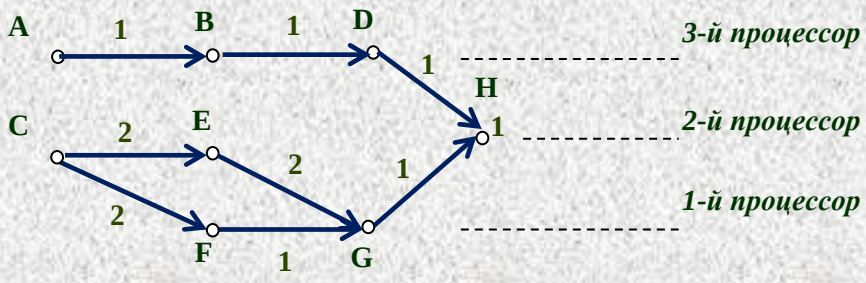


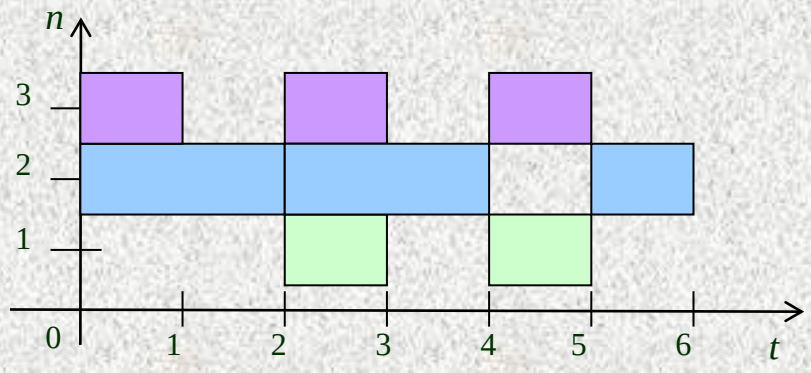
✓ Насколько непрерывно работают все три задействованные в вычислительном процессе системе? Сколько времени простаивает каждый процессор в процессе работы алгоритма?

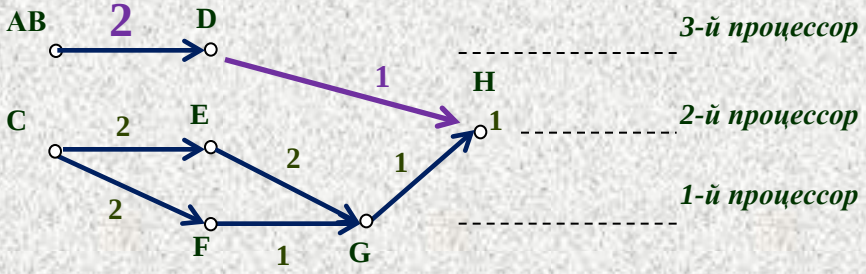
✓ Можно ли сократить общее время работы алгоритмы путем уменьшения времени простоев процессоров?

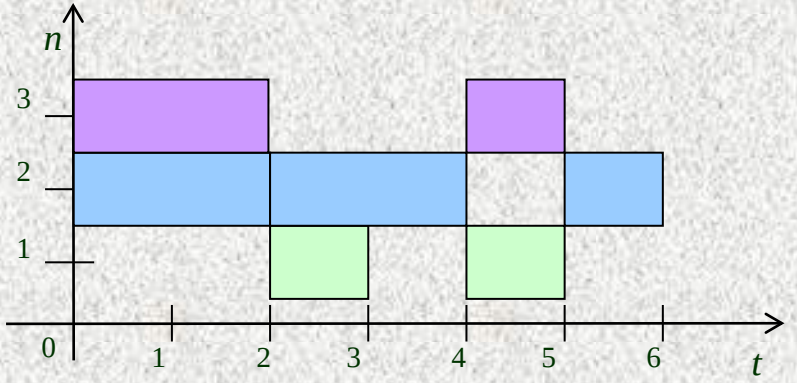
✓ Можно ли сократить ширину алгоритму путем уменьшения времени простоев процессоров за счет объединения мелких (по времени работы) операций в более крупные?

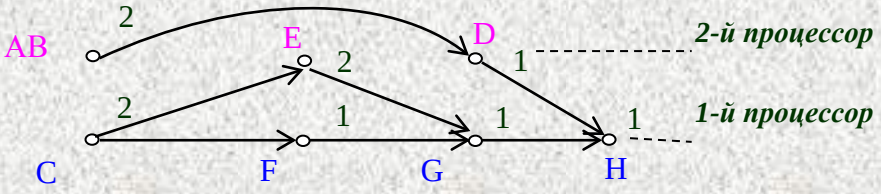
Оптимизация алгоритма по времени

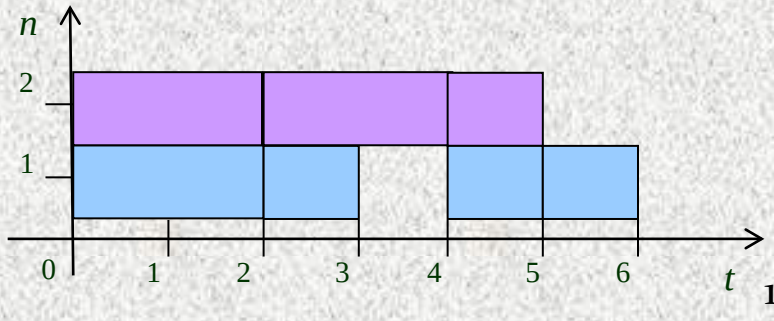




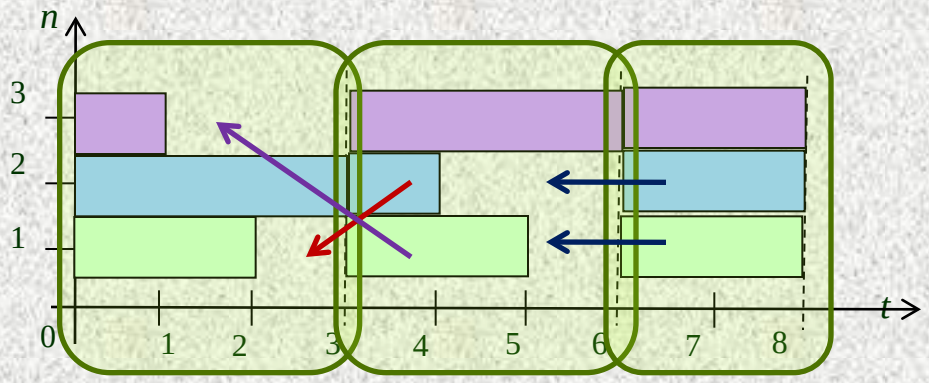








Оптимизация алгоритма по времени



1. $t_0=0$
2. $\max(t_{1j})_{j=1..3}=3 \Rightarrow t_0= t_0 +3$

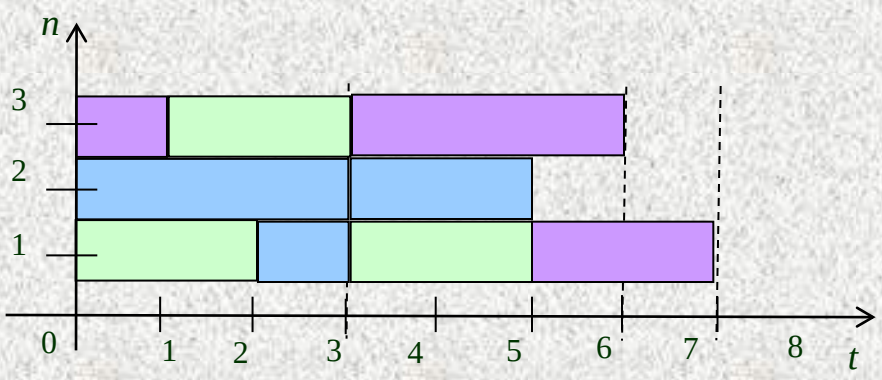
   j=1: $t_{11}=2$, $\Delta t=\max-t_{11}=3-2=1$

   $1=t_{22} \Rightarrow t_{11}= (t_{11}, t_{22})$, $t_{22} =0$

   j=2: $t_{12}=3$, $\Delta t=\max-t_{12}=3-3=0$

   j=3: $t_{13}=1$, $\Delta t=\max-t_{13}=3-1=2$

   $2=t_{21} \Rightarrow t_{13}= (t_{21}, t_{13})$, $t_{21} =0$



3. $\max(t_{2j})_{j=1..3}=3 \Rightarrow t_0=t_0+3=6$

   j=1: $t_{21}=0$, $\Delta t=\max-t_{21}=3-0=3$

   $3>t_{3i}$ (i=1..3) $\Rightarrow$

   $t_{21}= (t_{21}, t_{31})$, $t_{31} =0$

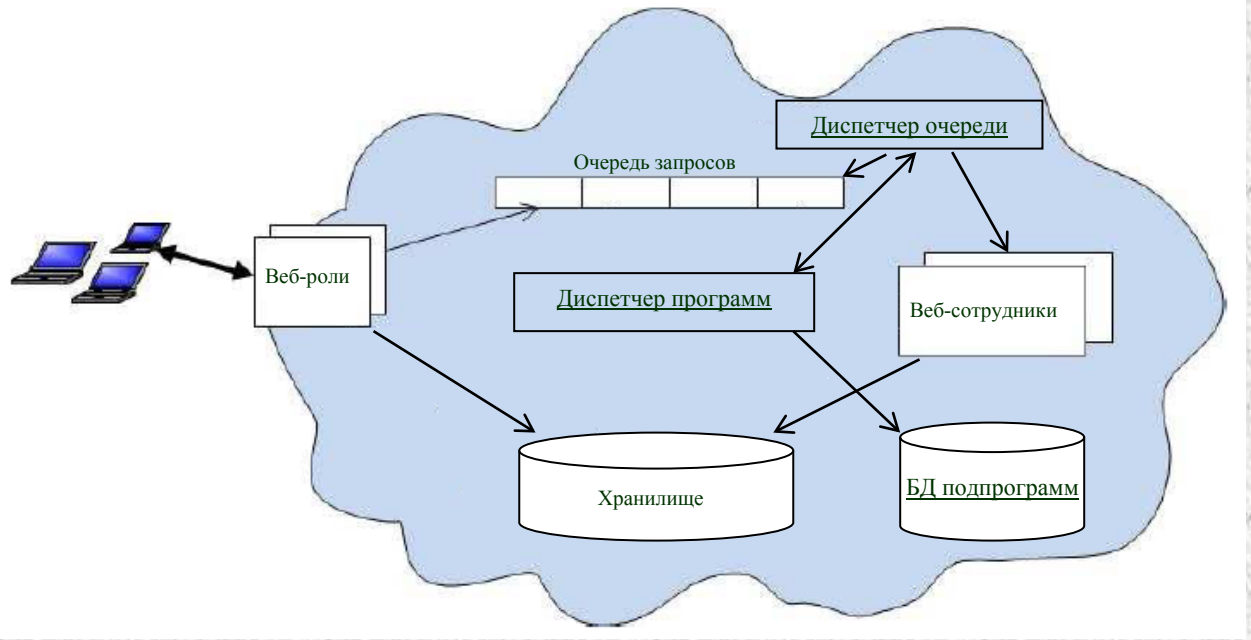
   j=2: $t_{22}=0$, $\Delta t=\max-t_{22}=3-0=3$

   $3>t_{3i}$ (i=1..3) $\Rightarrow$

   $t_{22}= (t_{22}, t_{32})$, $t_{32} =0$

   j=3: $t_{23}=3$, $\Delta t=\max-t_{23}=3-3=0$

Облачные вычисления

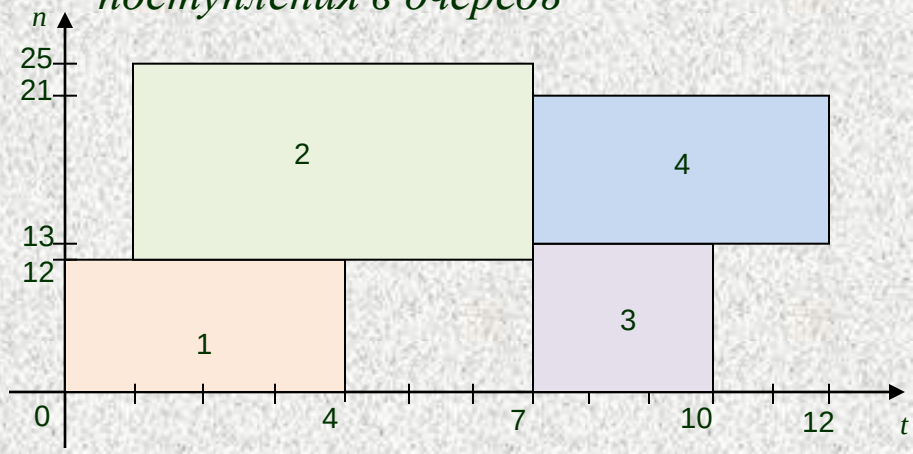


ЗАДАЧИ

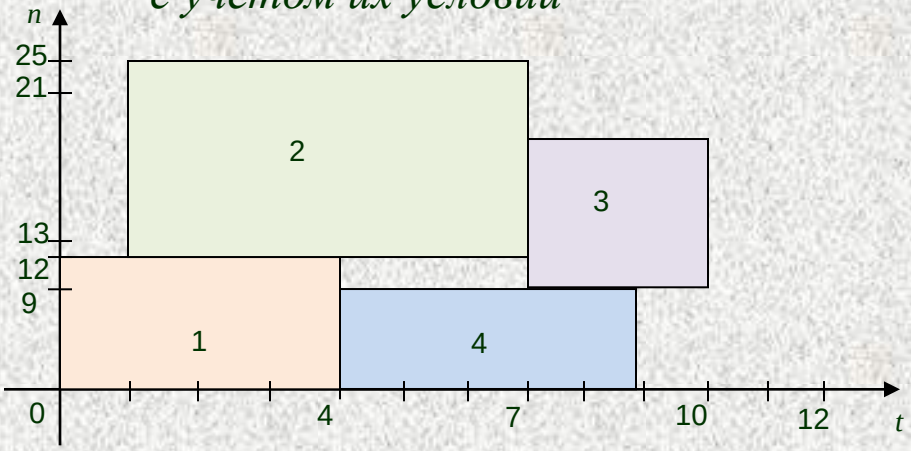
- распараллеливание отдельных программ;
- определение числа ролей-сотрудников и времени, которое они будут заняты;
- проектирование структуры базы данных и очереди запросов.

Схемы решения задач в облаке

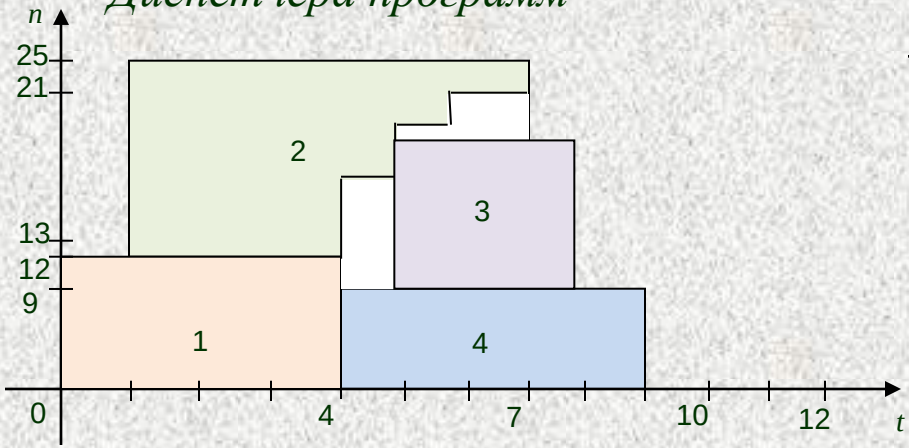
1 *Схема решения задач по мере их поступления в очередь*



2 *Схема решения задач с учетом их условий*

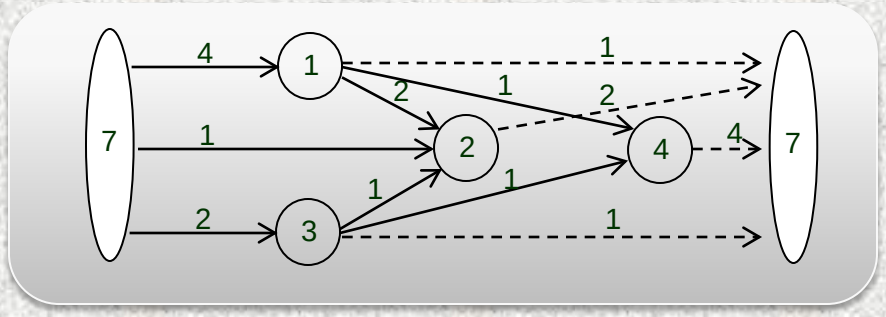


3 *Схема решения задач при наличии Диспетчера программ*



Графовые модели организации облака

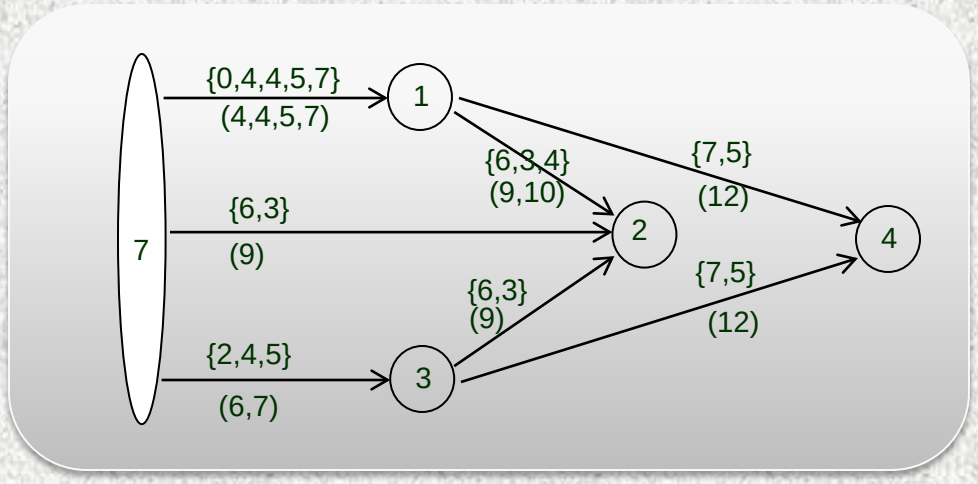
Модель организации без учета времени



Параметр «Процесс» решения *j*-й задачи на *n* процессорах

$$P\{t_j, t_{11}, \ldots, t_{i1}, \ldots, t_{n1}\}$$

Модель организации с учетом времени



Параметр «Сроки» решения *j*-й задачи на *n* процессорах

$$S\{t_j + t_{11}, \ldots, t_j + t_{i1}, \ldots, t_j + t_{n1}\}$$

Графовый алгоритм распределения задач

1. Первой задаче, поступившей в очередь, ставится в соответствие узел с номером задачи и ребро с параметром «Процесс» *P*.

2. Следующей задаче, поступившей в очередь, ставится в соответствие узел с номером задачи. Если:

- $N_c \geq n_j$, то добавляется ребро с весами *P* и *S*, где N_c - число свободных процессоров;
- $N_c < n_j$, то добавляем ребро с весом пустого множества *P{}* ;
- $N_c = 0$, то ребро не добавляем.

3. Если в очереди нет задач, то шаг 2. Иначе решается локальная задача:

Пусть t_{i4}, t_{j4} – время поступления *i*-й и *j*-й задач в очередь (*i*<*j*); m_i – число недостающих вычислительных ресурсов для *i*-й задачи; n_j – число процессоров, необходимых для решения *j*-й задачи; $\tau_{n_j k}$ – время работы *k*-го процессора, задействованногов решении *j*-й задачи; $R = S_0 \cup S_1 \cup \ldots \cup S_j$ – отсортированный параметр «Ресурсы». Тогда запуск *j*-й задачи ранее *i*-й задачи возможен в случае, если выполняется условие:

$$t_{j4} - t_{i4} + R_{m_i} + \max_{1 \leq k \leq n_i}(\tau_{n_i k}) \leq t_{i3}$$

Если запуск j-й задачи невозможен, то на графе для добавления i-й задачи следует вычеркнуть из множества R первые m_i чисел и провести ребра к узлу с номером *i* от тех узлов, во множестве *S* которых есть вычеркнутые числа. Подписываем над ребрами соответствующие расписания. Время начала решения задачи будет вычисляться по правилу: $t_i = R_{m_i}$

Формализованное распределение задач в облачном кластере

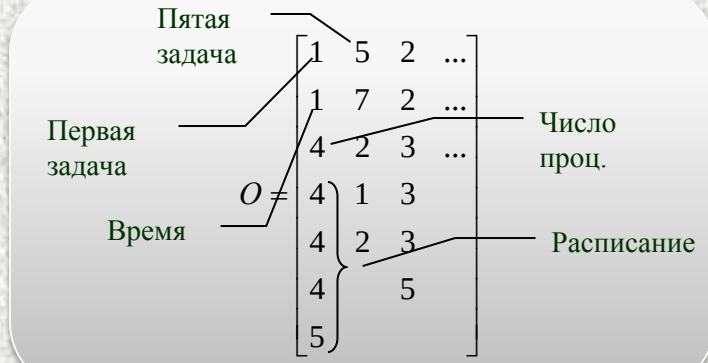
Пусть:

- облачный кластер содержит N процессоров;
- массив «Очередь» $O(M \times (N+3))$ содержит информацию о номерах задач, где M – размер очереди;
- массив «Процесс» $P(N \times 3)$ содержит информацию о номерах вычисляемых в настоящее время на облачном кластере задач;

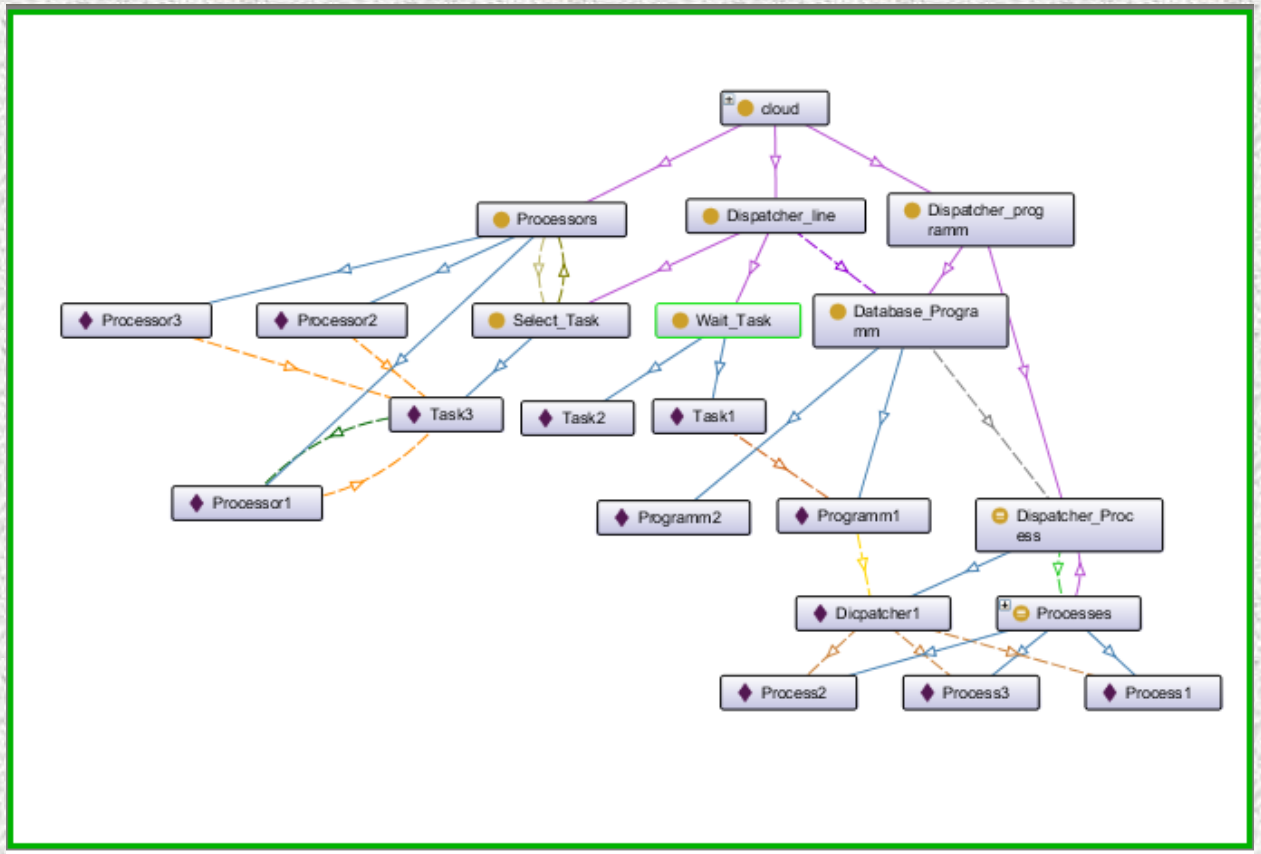
Значение S вычисляется по правилу: $S_j = t + P_{2j}$, где $j = \overline{1; N}$, t – время начала решения задачи.

- величина K – число занятых в текущий момент процессоров.

$$P = \begin{bmatrix} 1 & 1 & i & i & k & 1 & k \\ \tau_{11} & \tau_{12} & \tau_{i3} & \tau_{i4} & \tau_{k5} & \tau_{16} & \tau_{k7} \\ S_1 & S_2 & S_3 & S_4 & S_5 & S_6 & S_7 \end{bmatrix}$$



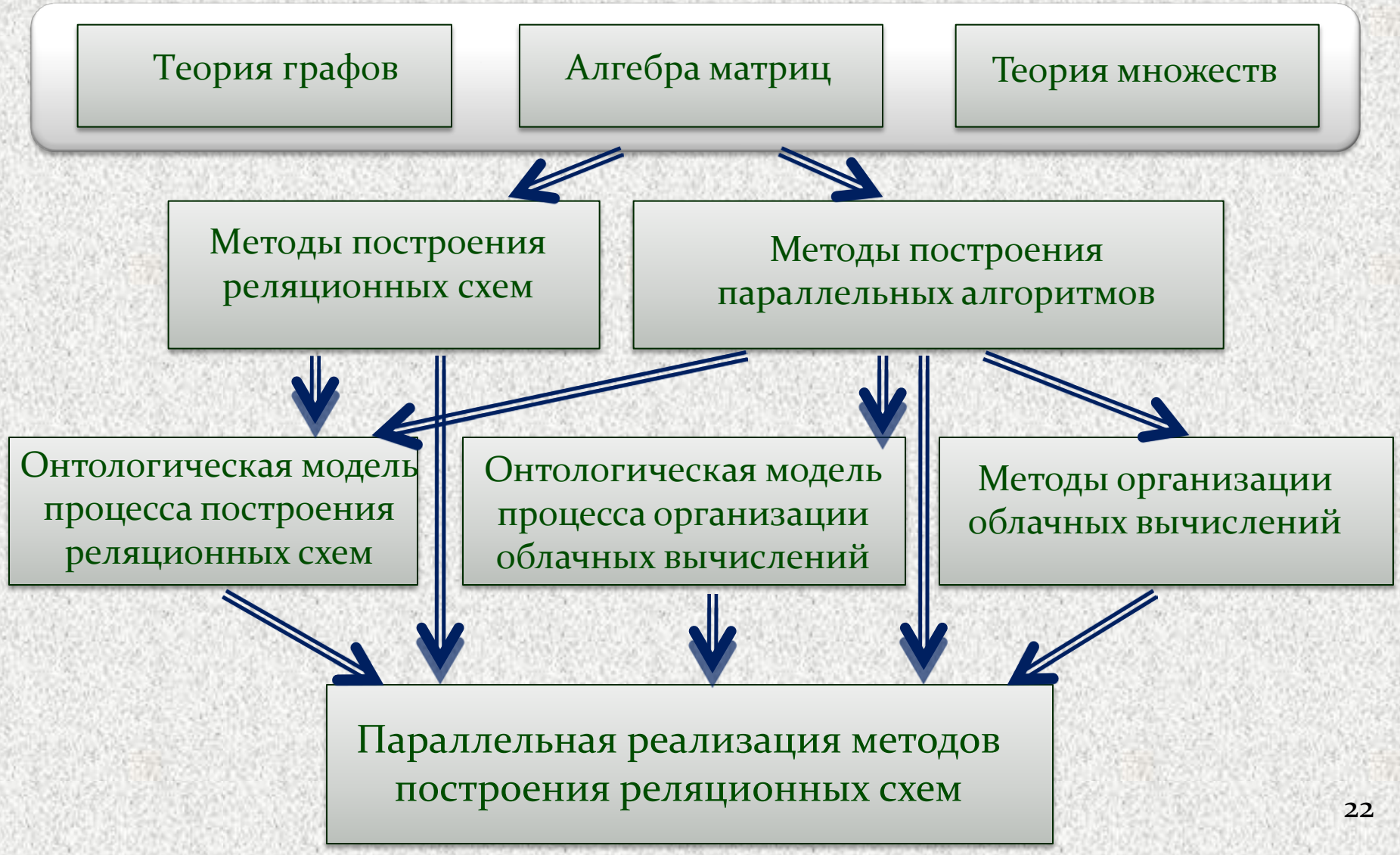
Онтологическая модель «облачного кластера»



dl-запросы

№	Запрос	Описание	Результат
1.	Database_Programm and hasDispatcherProcess value Dicpatcher1	Какая программа выполняется в соответствии с расписанием Dicpatcher1	Programm1
2.	Database_Programm and hasDispatcherProcess some (Dispatcher_Process and Number_Process value 1)	Какая программа выполняется в соответствии с процессом 1	Programm1
3.	Dispatcher_Process and hasDispatcherProcess some (hasProgramm some (Number_Task value 1))	По какому расписанию должна выполняться задача 1	Task1
4.	Dispatcher_line and not (Wait_Task)	Вывести список выполняемых задач	Task3
5.	Wait_Task and Enter_Time value "2012-02-08T15:00:00"^^dateTime	Время поступления в очередь какой задачи равно "2012-02-08T15:00:00"	Task1
6.	Processors and solveTask some (Select_Task and Start_Time value "2012-01-08T16:00:00"^^dateTime)	Какие процессоры начали работу в "2012-01-08T16:00:00"	Processor1 Processor2 Processor3

Система моделирования реляционных схем



Правила преобразования орграфа в дерево

Правило 1.

Если из узла А в узел В существует несколько путей: $A \to A_{11} \to \ldots \to A_{1n} \to B$ и $A \to A_{21} \to \ldots \to A_{2m} \to B$, то следует удалить из орграфа дугу:

$$\begin{cases} (A_{1n}, B), & m < n \\ (A_{1m}, B), & n < m \end{cases}$$

Правило 2.

Если из узла А в узел В существует несколько равных путей, B_1 и B_2 – предпоследние узлы в данных путях, то следует удалить из орграфа дугу:

$$\begin{cases} (B_1, A), d^-(B) = 1 \cap d^-(B_1) = 1 \cap d^-(B_2) > 1 \\ (B_1, A) \text{ или } (B_2, B), \text{ в остальных случаях} \end{cases}$$

Правило 3.

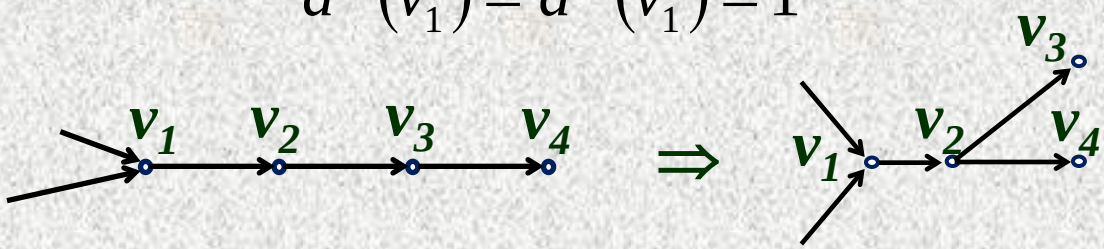
Если между узлами А и В существует контур: А→В, В→А, то следует удалить из орграфа дугу:

$$\begin{cases} (A, B), d^+(A) = d^-(B) = 0 \\ (A, B) \text{ или } (B, A), \text{ в остальных случаях} \end{cases}$$

Стягивание дерева

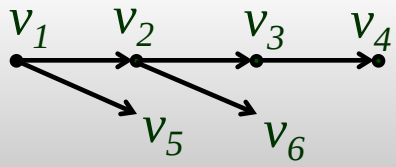
Стягивание дерева – процесс преобразования фрагмента дерева $(v_1, v_2, \ldots, v_n)$ в поддерево путем соединения нескольких последовательно идущих ребер в одной начальной вершине v_1, для которой выполняется правило: .

$$d^-(v_1) = d^+(v_1) = 1$$

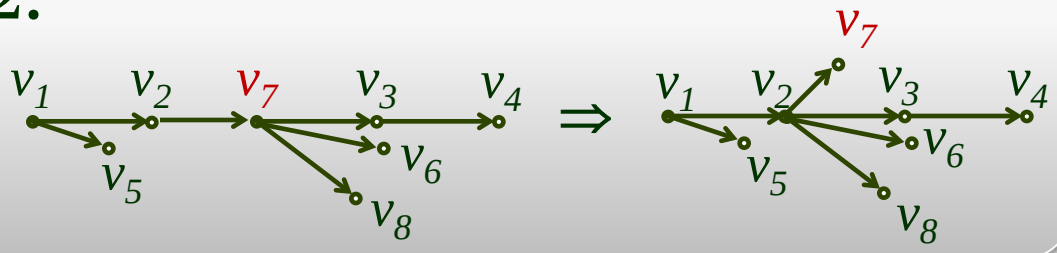


П р и м е р:

1.



2.



Пример

Заказы (№заказа, №заказчика, имя заказчика, улица, город, штат, индекс, №фильма, название фильма, количество, срок)

FD1: №заказа → №заказчика, срок;

FD2: №заказчика → имя;

FD3: №заказчика → улица, индекс;

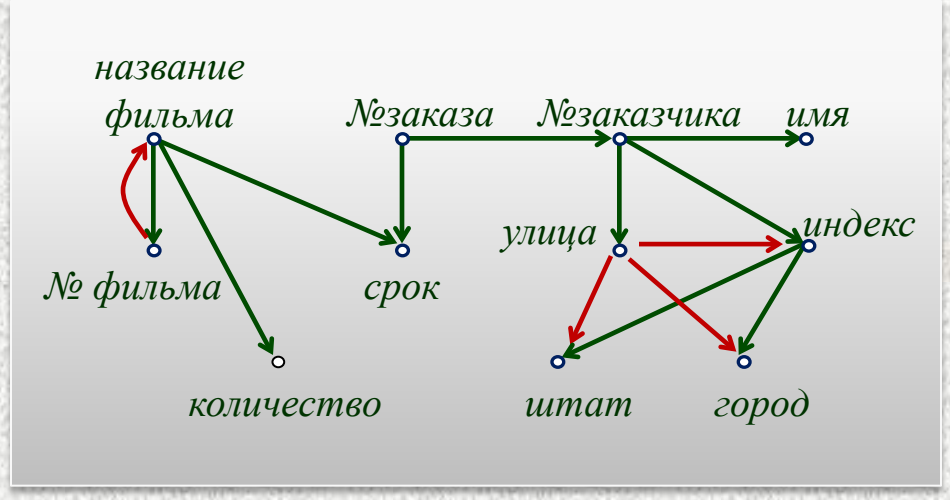
FD4: индекс → штат, город;

FD5: улица → город, штат, индекс;

FD6: №фильма → название фильма;

FD7: название фильма → количество, срок;

FD8: название фильма → №фильма;

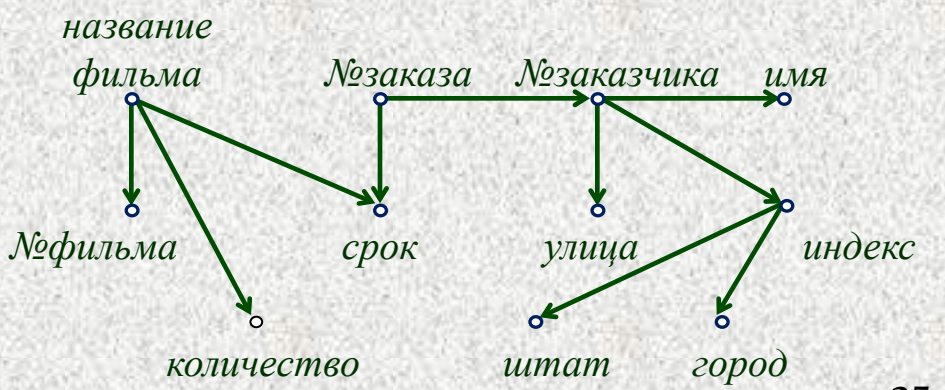


S1 (№заказа, №фильма, название фильма, количество);

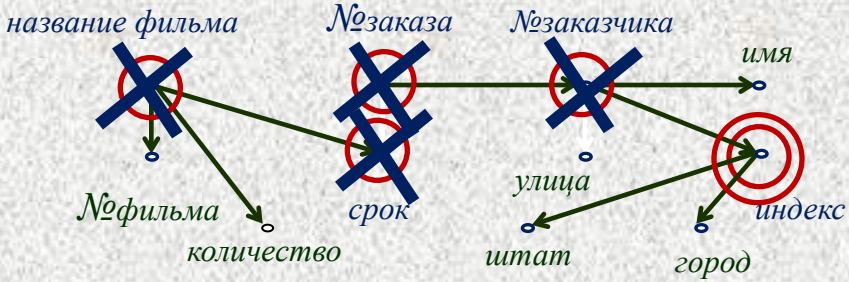
S2 (№заказа, №заказчика, срок);

S3 (№заказчика, имя, улица, индекс).

S4 (индекс, город, штат);



Выбор точки декомпозиции



Точка сочленения орграфа – узел, удаление которого приведет к делению орграфа минимум на два связных орграфа.

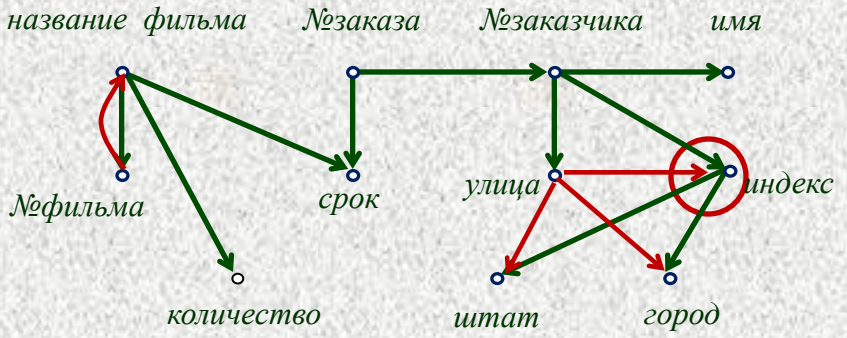
Точка декомпозиции графа – узел, соответствующий атрибуту, который является граничным при делении отношения R=R1∪R2, т.е. входит в качестве первичного ключа в отношение R1 и внешнего ключа в отношение R2.

Точка сочленения является точкой декомпозиции, если она удовлетворяет следующим условиям:

1. $d^+(v_1) \geq 1$, т.е. является хотя бы для одной дуги конечным узлом;
2. $d^-(v_1) > 1$, т.е.является вершиной дерева орграфа функциональных зависимостей.
3. Поддерево с вершиной в точке декомпозиции является концевым поддеревом орграфа и не содержит в себе других поддеревьев.

Преобразование орграфа в дерево

Строка декомпозиции матрицы смежности A – i-я строка, для которой выполняется условие: $R_i \cap S_j = \varnothing$ для всех j: $A_{ij} \neq 0$, где R – множество атрибутов, входящих в название i-й строки, S – множество атрибутов, входящих в названия столбцов, на пересечении которых с i-й строкой стоят единицы.



	№з	№з-ка	имя	улица	город	штат	индекс	№ф	Наз.ф	Кол-во	срок
№з	0	1	0	0	0	0	0	0	0	0	1
№з-ка	0	0	1	1	0	0	1	0	0	0	0
индекс	0	0	0	0	1	1	0	0	0	0	0
улица	0	0	0	0	1	1	1	0	0	0	0
№ф	0	0	0	0	0	0	0	0	1	0	0
Наз. ф	0	0	0	0	0	0	0	1	0	1	1

$$\{индекс\} \cap \{город, штат\} = \varnothing$$

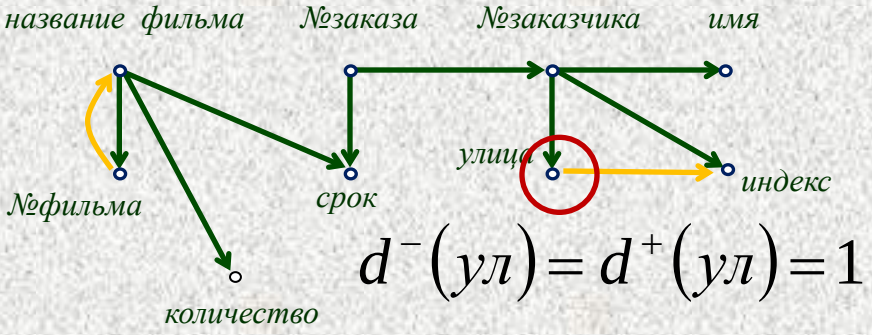
S1 (индекс, город, штат);

S2 (индекс, №фильма, название фильма, №заказа, №заказчика, срок, имя, количество, улица);

Преобразование орграфа в дерево

S1 (индекс, город, штат);

S2 (индекс, №фильма, название фильма, №заказа, №заказчика, срок, имя, количество, улица);



$$d^-(ул) = d^+(ул) = 1$$

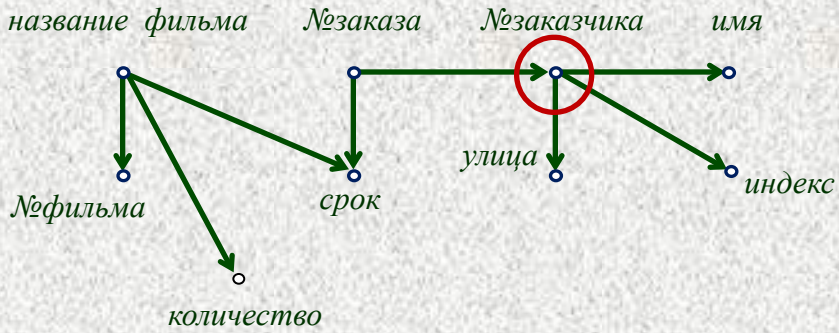
$$\sum A_{i,ул} = 1$$

	№з	№з-ка	Имя	улица	индекс	№ф	Наз.ф	Кол-во	срок
№з	0	1	0	0	0	0	0	0	1
№з-ка	0	0	1	1	1	0	0	0	0
улица	0	0	0	0	1	0	0	0	0
№ф	0	0	0	0	0	0	1	0	0
Наз. ф	0	0	0	0	0	1	0	1	1

$$\sum A_{ул,j} = 1$$

Выбор точки декомпозиции

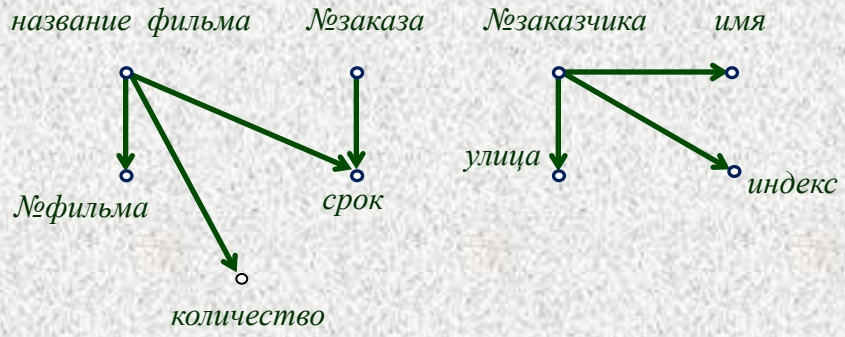
S1 (индекс, город, штат);
S2 (индекс, №фильма, название фильма, №заказа, №заказчика, срок, имя, количество, улица);



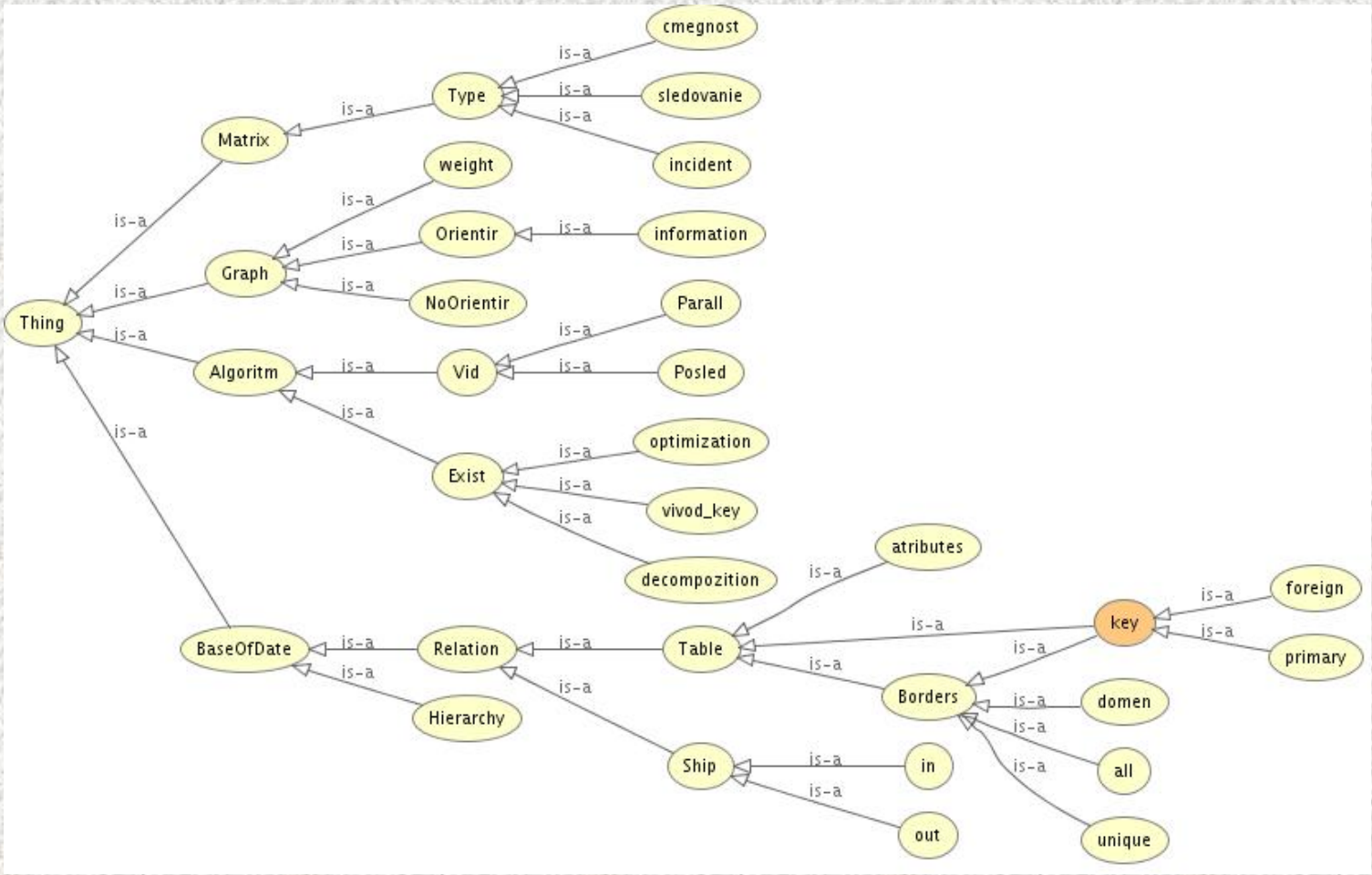
$$\{№з, №з\text{-}ка, Наз.ф\} \cap \{Имя, улица, индекс\} = \varnothing$$

	№з	№з-ка	Имя	улица	индекс	№ф	Кол-во	срок
№з	0	1	0	0	0	0	0	1
№з-ка	0	0	1	1	1	0	0	0
Наз. ф	0	0	0	0	0	1	1	1

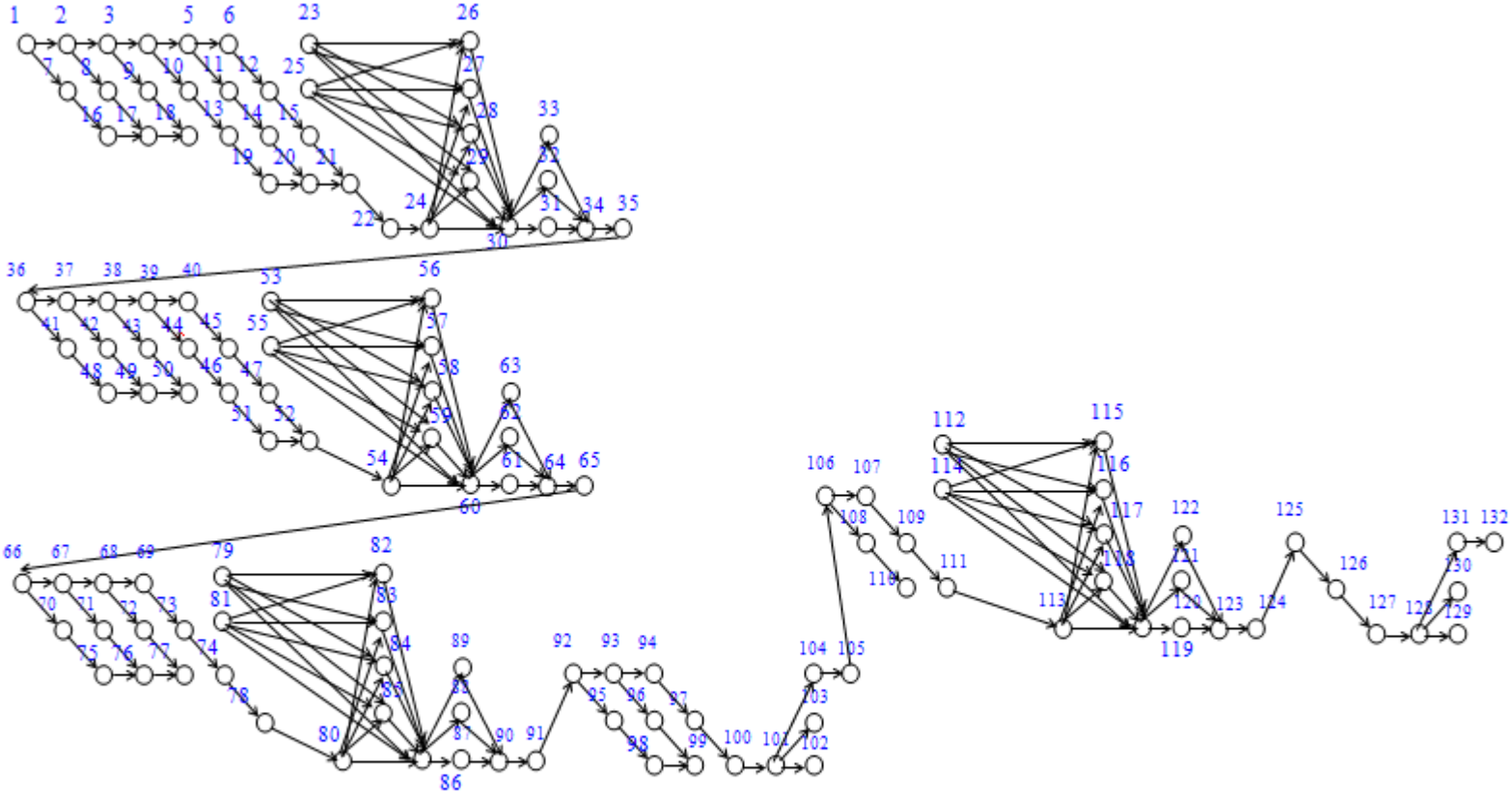
S1 (индекс, город, штат);
S2 (№заказчика, имя, улица, индекс);
S3 (№фильма, название фильма, №заказа, №заказчика, срок, количество).



Онтологическая модель процесса построения реляционных схем



Распараллеливание алгоритма нормализации реляционных отношений



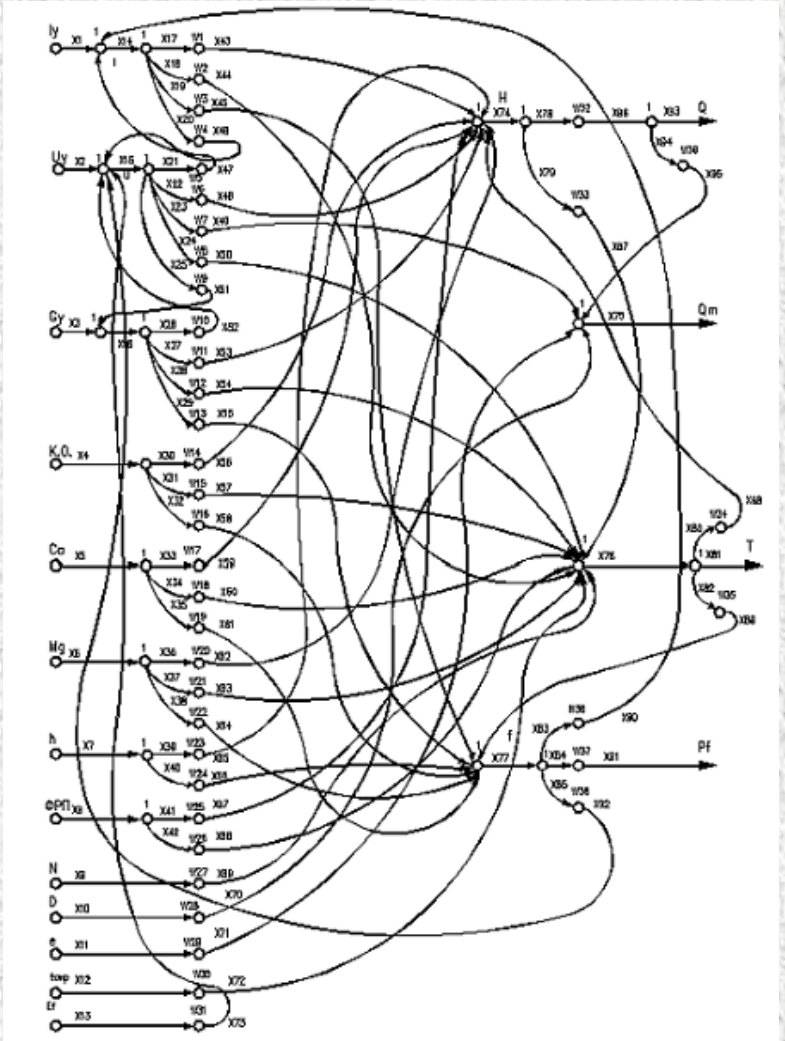
Информационный граф алгоритма нормализации реляционных отношений

Таблица функциональных связей исследуемых технологических параметров.

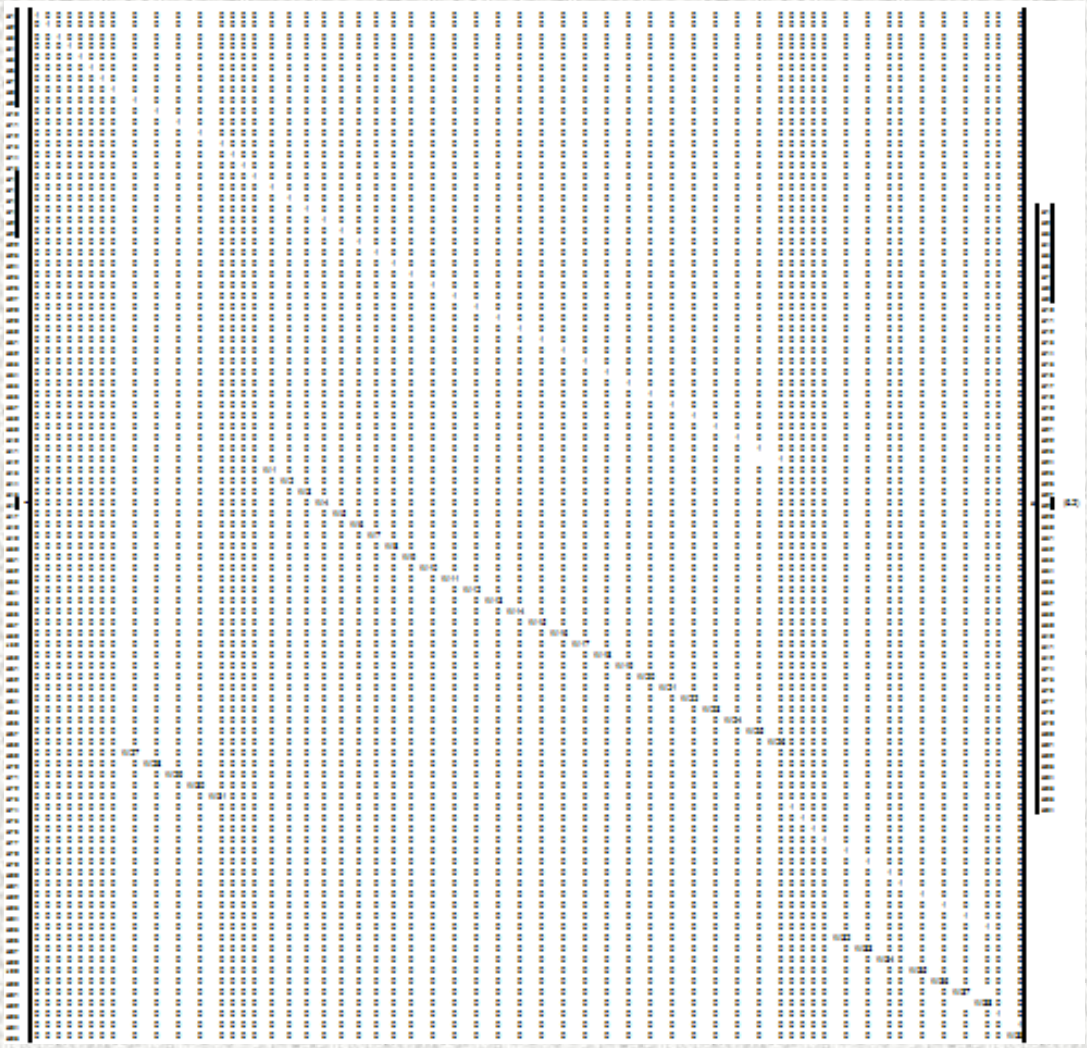
Исследуемые технологические параметры	I	U	G	H	Q	Оэ	T	f
Iy								
Uy								
Gy								
К.О.								
CaF_2								
MgF_2								
h								
ФРП								
N								
D								
e								
tокр								
Ef								
I								
U								
G								
H								
Q								
T								
f								

Структурная идентификация процесса получения алюминия

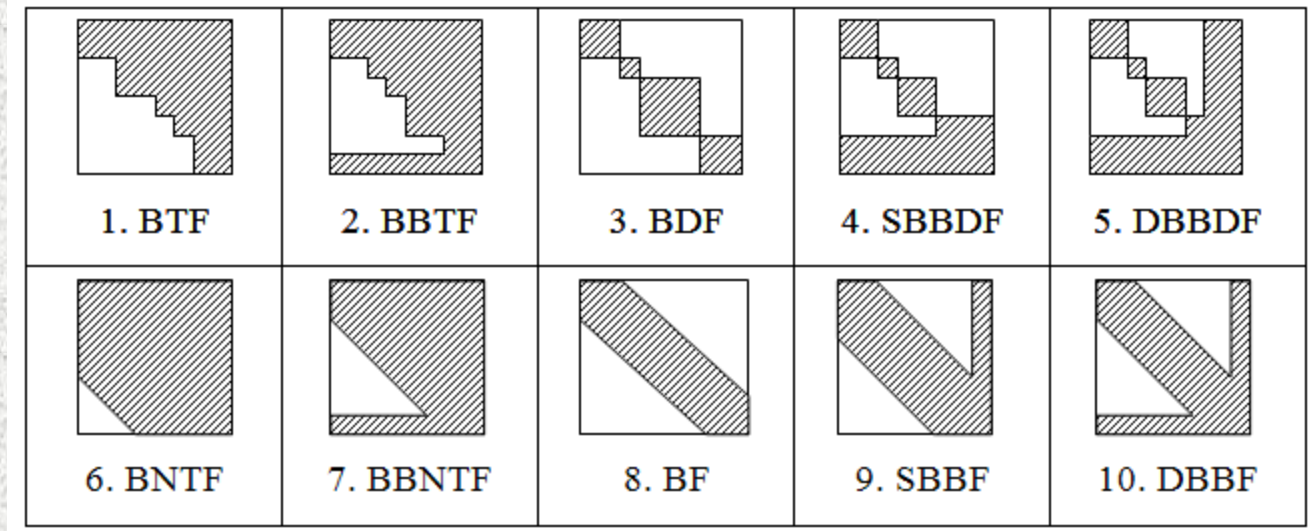
$x_1=1*x_1$; $x_{33}=1*x_{33}$; $x_{65}=W_{23}*x_{39}$;
$x_2=1*x_2$; $x_{34}=1*x_{34}$; $x_{66}=W_{24}*x_{40}$;
$x_3=1*x_3$; $x_{35}=1*x_{35}$; $x_{67}=W_{25}*x_{41}$;
$x_4=1*x_4$; $x_{36}=1*x_{36}$; $x_{68}=W_{26}*x_{42}$;
$x_5=1*x_5$; $x_{37}=1*x_{37}$; $x_{69}=W_{27}*x_9$;
$x_6=1*x_6$; $x_{38}=1*x_{38}$; $x_{70}=W_{28}*x_{10}$;
$x_7=1*x_7$; $x_{39}=1*x_{39}$; $x_{71}=W_{29}*x_{11}$;
$x_8=1*x_8$; $x_{40}=1*x_{40}$; $x_{72}=W_{30}*x_{12}$;
$x_9=1*x_9$; $x_{41}=1*x_{41}$; $x_{73}=W_{31}*x_{13}$;
$x_{10}=1*x_{10}$; $x_{42}=1*x_{42}$; $x_{74}=1*x_{74}$;
$x_{11}=1*x_{11}$; $x_{43}=W_1*x_{17}$; $x_{75}=1*x_{75}$;
$x_{12}=1*x_{12}$; $x_{44}=W_2*x_{18}$; $x_{76}=1*x_{76}$;
$x_{13}=1*x_{13}$; $x_{45}=W_3*x_{19}$; $x_{77}=1*x_{77}$;
$x_{14}=1*x_{14}$; $x_{46}=W_4*x_{20}$; $x_{78}=1*x_{78}$;
$x_{15}=1*x_{15}$; $x_{47}=W_5*x_{21}$; $x_{79}=1*x_{79}$;
$x_{16}=1*x_{16}$; $x_{48}=W_6*x_{22}$; $x_{80}=1*x_{80}$;
$x_{17}=1*x_{17}$; $x_{49}=W_7*x_{23}$; $x_{81}=1*x_{81}$;
$x_{18}=1*x_{18}$; $x_{50}=W_8*x_{24}$; $x_{82}=1*x_{82}$;
$x_{19}=1*x_{19}$; $x_{51}=W_9*x_{25}$; $x_{83}=1*x_{83}$;
$x_{20}=1*x_{20}$; $x_{52}=W_{10}*x_{26}$; $x_{84}=1*x_{84}$;
$x_{21}=1*x_{21}$; $x_{53}=W_{11}*x_{27}$; $x_{85}=1*x_{85}$;
$x_{22}=1*x_{22}$; $x_{54}=W_{12}*x_{28}$; $x_{86}=W_{32}*x_{78}$;
$x_{23}=1*x_{23}$; $x_{55}=W_{13}*x_{29}$; $x_{87}=W_{33}*x_{79}$;
$x_{24}=1*x_{24}$; $x_{56}=W_{14}*x_{30}$; $x_{88}=W_{34}*x_{80}$;
$x_{25}=1*x_{25}$; $x_{57}=W_{15}*x_{31}$; $x_{89}=W_{35}*x_{82}$;
$x_{26}=1*x_{26}$; $x_{58}=W_{16}*x_{32}$; $x_{90}=W_{36}*x_{83}$;
$x_{27}=1*x_{27}$; $x_{59}=W_{17}*x_{33}$; $x_{91}=W_{37}*x_{84}$;
$x_{28}=1*x_{28}$; $x_{60}=W_{18}*x_{34}$; $x_{92}=W_{38}*x_{85}$;
$x_{29}=1*x_{29}$; $x_{61}=W_{19}*x_{35}$; $x_{93}=1*x_{93}$;
$x_{30}=1*x_{30}$; $x_{62}=W_{20}*x_{36}$; $x_{94}=1*x_{94}$;
$x_{31}=1*x_{31}$; $x_{63}=W_{21}*x_{37}$; $x_{95}=W_{38}*x_{94}$;
$x_{32}=1*x_{32}$; $x_{64}=W_{22}*x_{38}$;



Структурная идентификация процесса получения алюминия



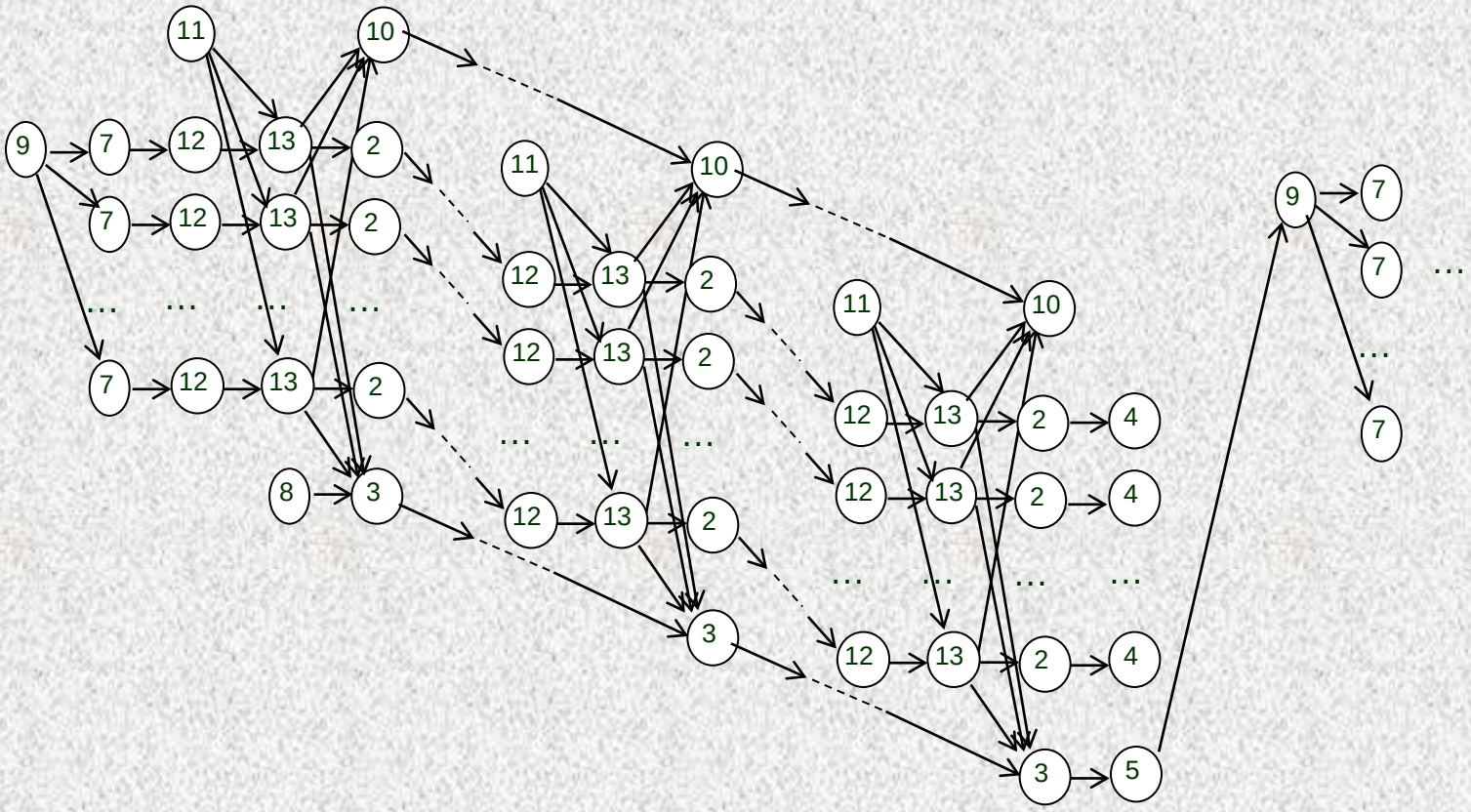
Схемы разреженных матриц



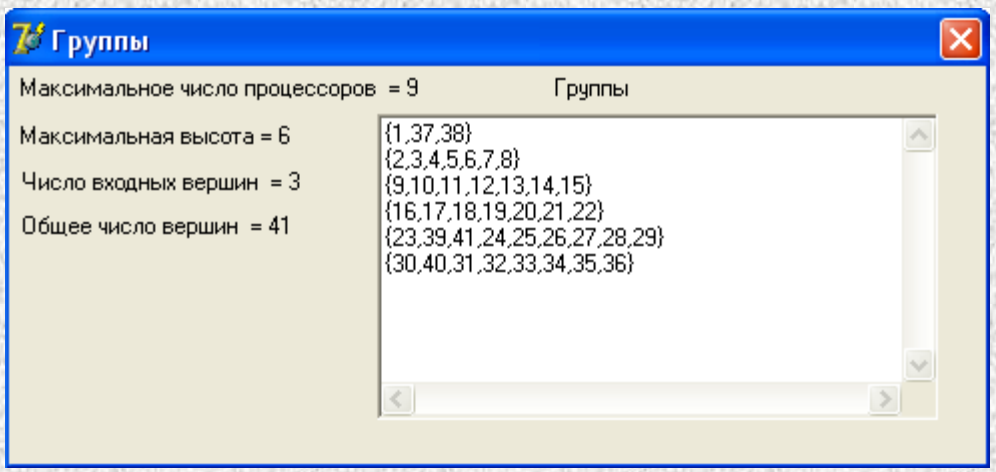
$$\begin{pmatrix} A_1 & & & 0 \\ & A_2 & & \\ & & \ddots & \\ 0 & & & A_k \\ & A_{k+1} & & \end{pmatrix} \cdot \begin{pmatrix} X_1 \\ X_2 \\ \vdots \\ X_k \end{pmatrix} = \begin{pmatrix} B_1 \\ B_2 \\ \vdots \\ B_k \\ B_{k+1} \end{pmatrix} \quad (1)$$

$$\begin{cases} A_1 X_1 = B_1 \\ A_2 X_2 = B_2 \\ \quad \cdots\cdots \\ A_k X_k = B_k \\ A_{k+1}(X_1, X_2, \ldots, X_k) = B_{k+1} \end{cases} \quad (2)$$

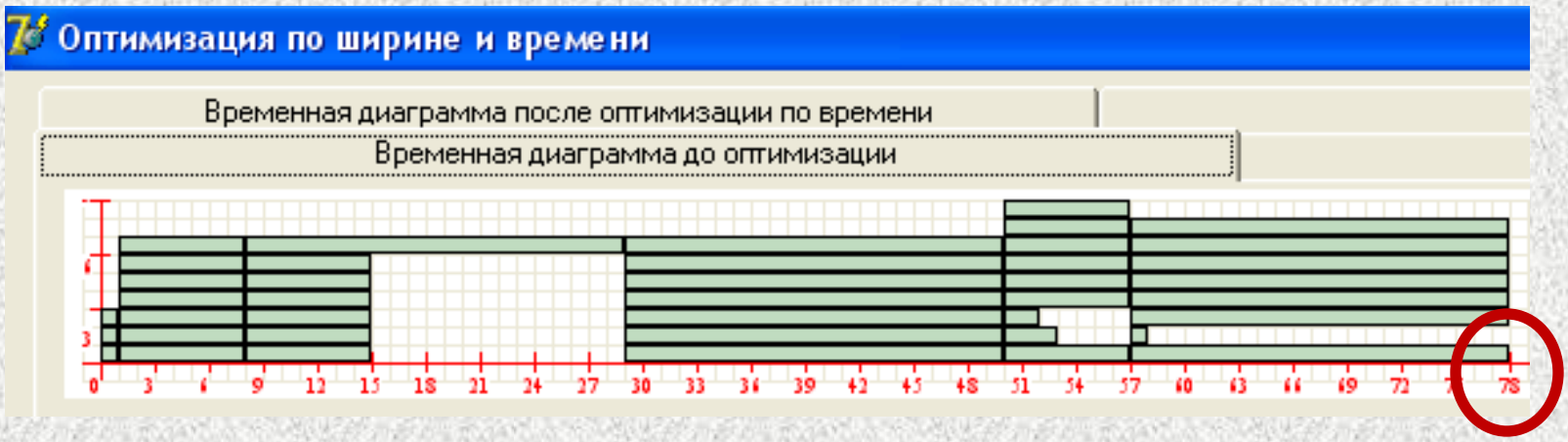
Структурная идентификация процесса получения алюминия



Структурная идентификация процесса получения алюминия



T=78ед.



Структурная идентификация процесса получения алюминия

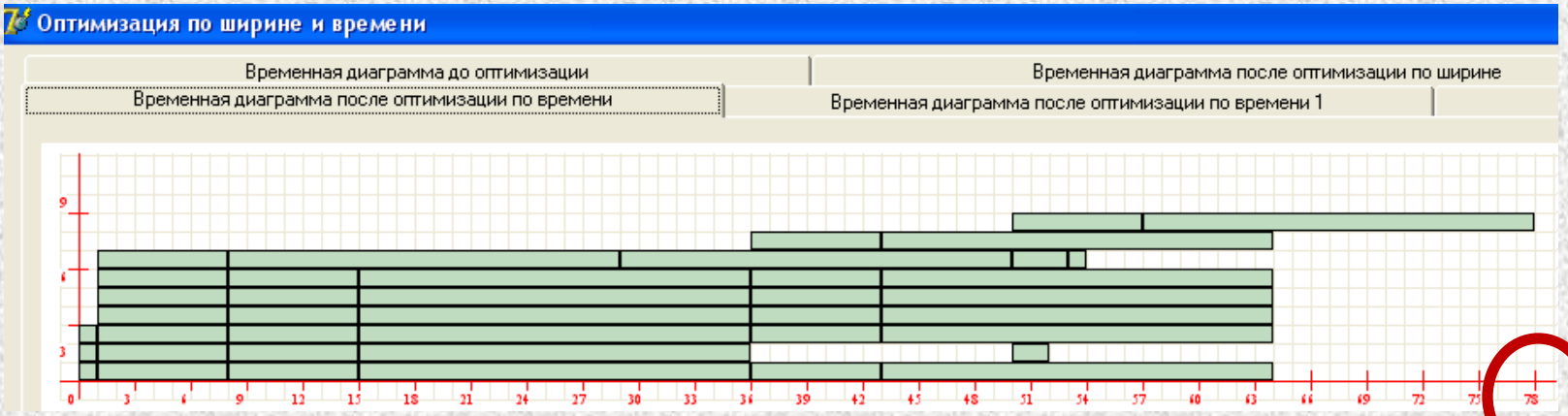


Рис.1

Т=78ед.

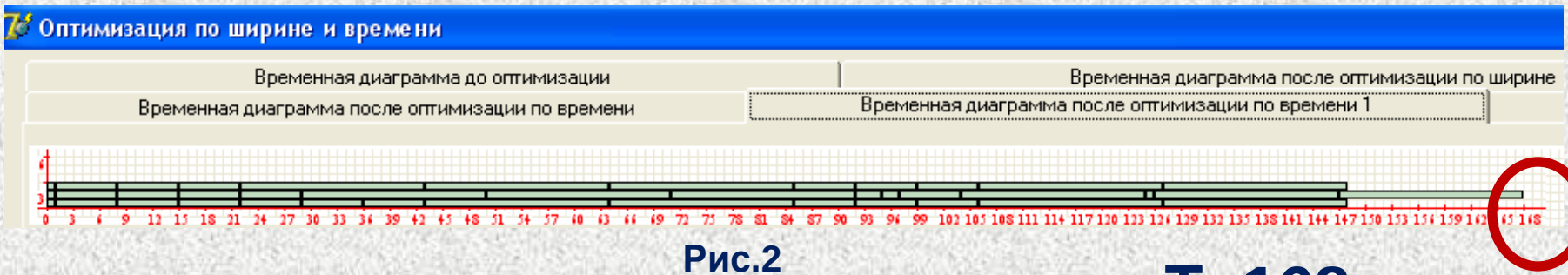


Рис.2

Т=168ед.

Заключение

1. Предложено представление функциональных зависимостей реляционных схем в матричном виде, и получены матричные аналоги свойств функциональных зависимостей.

2. Разработан метод построения реляционной схемы на основе атрибутов предметной области и декларированных функциональных зависимостей. Все получаемые схемы удовлетворяют нормальной форме Бойса-Кодда.

3. Разработан метод нахождения первичного ключа отношения на основе функциональных зависимостей.

5. Разработан метод приведения разреженной матрицы к треугольному виду с сохранением первоначальных значений элементов.

6. Разработан метод оптимизации информационного графа параллельного алгоритма по ширине, т.е. по числу процессоров, задействованных в решении поставленной прикладной задачи. Метод состоит из двух частей, которые могут применяться независимо друг от друга.

Заключение

7. Получены оценки теоретической и практической минимальной ширины информационного графа, по которым можно принимать решение о дальнейшем его преобразовании.

8. Разработан метод оптимизации параллельного алгоритма по времени выполнения, позволяющий осуществить равномерную загрузку процессоров, сократить время решения поставленной задачи и уменьшить ширину информационного графа.

9. Разработан метод оптимизации информационного графа параллельного алгоритма по ширине, основанный на списках смежности. Метод состоит из двух частей, которые могут применяться независимо друг от друга.

10. Разработан метод оптимизации информационного графа параллельного алгоритма по ширине, основанный на списках следования. Метод обеспечивает равномерное заполнение групп и построен таким образом, что ширина каждого яруса будет приближена к теоретической минимальной ширине, никогда не превосходя ее.

11. Разработано программное обеспечение для апробации полученных методов и проведения дальнейших исследований в данной области.

Заключение

12. Разработана онтологическая модель процесса построения реляционных схем на распределенных вычислительных системах, оптимального по нескольким параметрам.

13. Разработан метод построения детализированного взвешенного ориентированного графа распределения задач по ролям-сотрудникам при организации параллельных вычислений в облачном кластере.

14. Разработан формализованный алгоритм распределения задач по ролям-сотрудникам при облачных вычислениях, обладающий следующими характеристиками: возможность легкого переноса на любой императивный язык программирования и большой запас внутреннего параллелизма.

15. С помощью программы Protege 4.0 построена онтологическая модель облачного кластера, отражающая взаимосвязи между его составляющими и позволяющая с помощью dl-запросов получить информацию, необходимую для дальнейшего принятия решения о распределении очередной задачи из очереди.