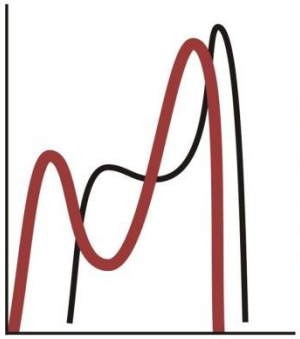


ИНСТИТУТ
МАТЕМАТИКИ
МЕХАНИКИ
КОМПЬЮТЕРНЫХ
НАУК

имени И.И. Воровича —

Архитектура компьютера и операционные системы

Лекция 21. Взаимоблокировки.

Андреева Евгения Михайловна
доцент кафедры информатики и вычислительного эксперимента

ИНСТИТУТ МАТЕМАТИКИ МЕХАНИКИ КОМПЬЮТЕРНЫХ НАУК имени И.И. Воровича

Взаимоблокировки

- Процесс 1
- Процесс 2

- запрос принтера
- запрос сканера

- запрос сканера
- запрос принтера

Основные определения

- Ресурс – любой объект, к которому предоставляется доступ (периферийные устройства, данные, файлы и т. д.)
- Тупик (**взаимоблокировка, deadlock**) – ситуация, когда два или несколько процессов не могут продолжаться, поскольку каждый из них ожидает освобождения ресурсов, занятых другим процессом.

ИНСТИТУТ МАТЕМАТИКИ МЕХАНИКИ КОМПЬЮТЕРНЫХ НАУК имени И.И. Воровича

Классификация ресурсов

- Выгружаемые
  - страницы памяти (на ПК)
- Невыгружамые
  - принтеры
  - сканеры
  - магнитные ленты
  - …

ИНСТИТУТ МАТЕМАТИКИ МЕХАНИКИ КОМПЬЮТЕРНЫХ НАУК имени И.И. Воровича

Условия возникновения взаимоблокировок

1. **Условие взаимного исключения**. Каждый ресурс либо выделен в данный момент только одному процессу, либо доступен.
2. **Условие удержания и ожидания.** Процессы, удерживающие в данный момент ранее выделенные им ресурсы, могут запрашивать новые ресурсы.
3. **Условие невыгружаемости.** Ранее выделенные ресурсы не могут быть принудительно отобраны у процесса.
4. **Условие циклического ожидания.** Должна существовать кольцевая последовательность из двух и более процессов, каждый из которых ожидает высвобождения ресурса, удерживаемого следующим членом последовательности.

Коффман (Coffman et al., 1971)

ИНСТИТУТ МАТЕМАТИКИ МЕХАНИКИ КОМПЬЮТЕРНЫХ НАУК имени И.И. Воровича

Граф Холта

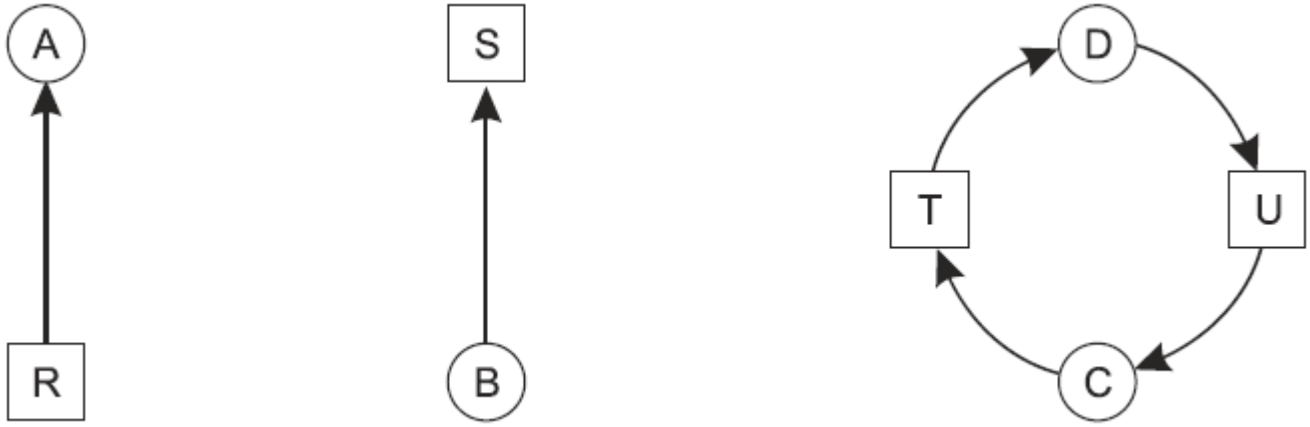
- Пусть в графе имеется два вида узлов
  Pi – процессы, Rj – ресурсы, $(2 \leq i \leq n, 2 \leq j \leq m)$

- Направленное ребро (Rj Pi) соответствует тому, что ресурс Rj удерживается процессом Pi

- Направленное ребро (Pi Rj) соответствует тому, что процесс Pi заблокирован в ожидании ресурса Rj

ИНСТИТУТ МАТЕМАТИКИ МЕХАНИКИ КОМПЬЮТЕРНЫХ НАУК имени И.И. Воровича

Примеры графа Холта



- Цикл в графе Холта означает наличие взаимоблокировки, включающей процессы и ресурсы (предполагается, что в системе есть только один ресурс каждого типа).

Пример взаимоблокировки

A

Запросить R
Запросить S
Освободить R
Освободить S

а

B

Запросить S
Запросить T
Освободить S
Освободить T

б

C

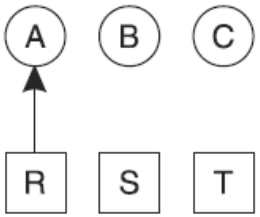
Запросить T
Запросить R
Освободить T
Освободить R

в

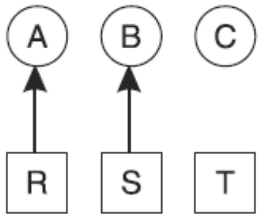
1. A запрашивает R
2. B запрашивает S
3. C запрашивает T
4. A запрашивает S
5. B запрашивает T
6. C запрашивает R

Взаимоблокировка

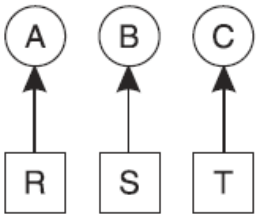
г



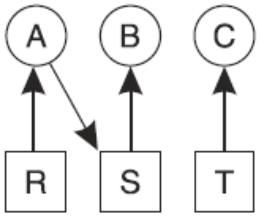
д



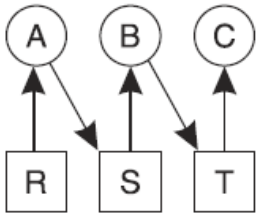
е



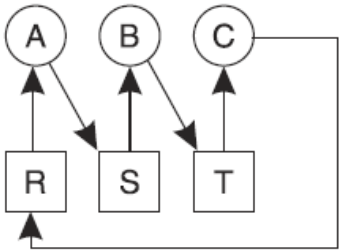
ж



з



и



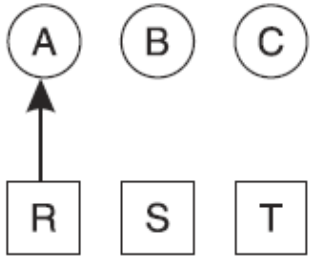
к

ИНСТИТУТ МАТЕМАТИКИ МЕХАНИКИ КОМПЬЮТЕРНЫХ НАУК имени И.И. Воровича

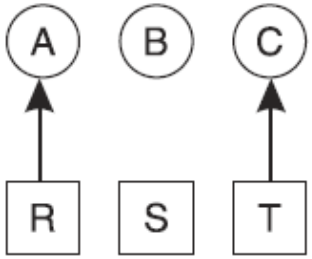
Пример предупреждения взаимоблокировки

1. A запрашивает R
2. C запрашивает T
3. A запрашивает S
4. C запрашивает R
5. A освобождает R
6. A освобождает S

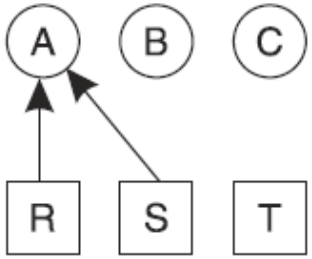
Нет взаимоблокировки



л

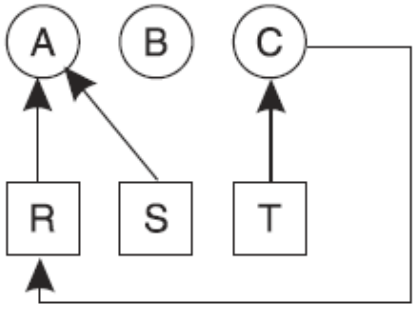


м

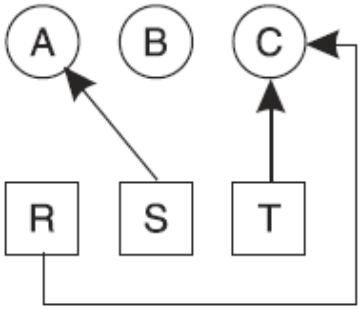


н

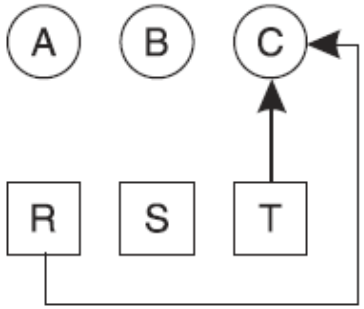
о



п



р



с

ИНСТИТУТ МАТЕМАТИКИ МЕХАНИКИ КОМПЬЮТЕРНЫХ НАУК имени И.И. Воровича

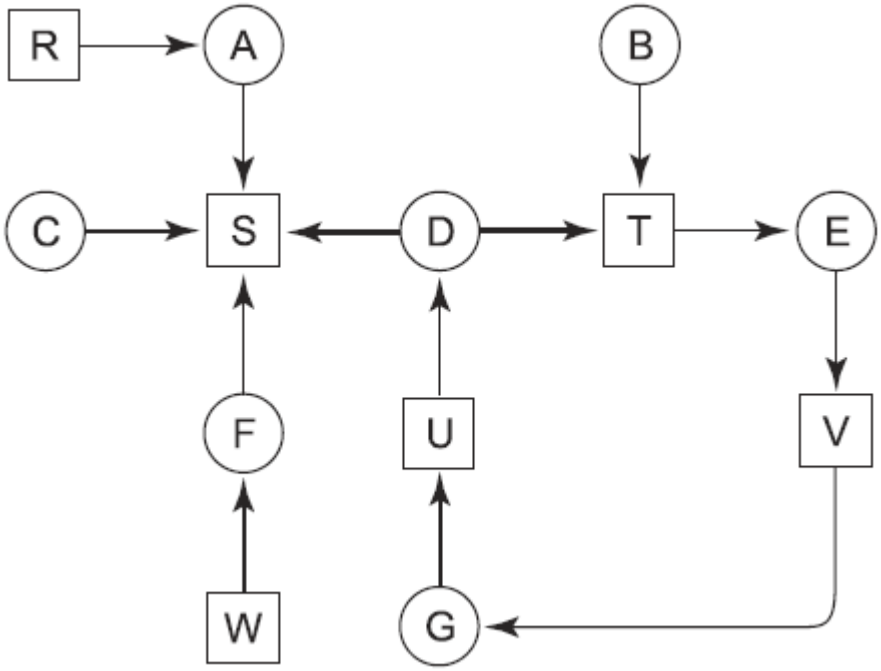
Стратегии борьбы с тупиками

- Игнорирование проблемы (страусиный алгоритм).
- Обнаружение и восстановление.
- Динамическое уклонение от них за счет тщательного распределения ресурсов.
- Предотвращение за счет структурного подавления одного из четырех условий, необходимых для их возникновения.

ИНСТИТУТ МАТЕМАТИКИ МЕХАНИКИ КОМПЬЮТЕРНЫХ НАУК имени И.И. Воровича

Обнаружение взаимоблокировки при использовании одного ресурса каждого типа

- **Поиск цикла в графе Холта**
- *A* удерживает *R* и хочет получить *S*.
- *B* не удерживает никаких ресурсов, но хочет получить *T*.
- *C* не удерживает никаких ресурсов, но хочет получить *S*.
- *D* удерживает *U* и хочет получить *S* и *T*.
- *E* удерживает *T* и хочет получить *V*.
- *F* удерживает *W* и хочет получить *S*.
- *G* удерживает *V* и хочет получить *U*.



ИНСТИТУТ МАТЕМАТИКИ МЕХАНИКИ КОМПЬЮТЕРНЫХ НАУК имени И.И. Воровича

Обнаружение взаимоблокировки при использовании нескольких ресурсов каждого типа

- Пусть в системе *n* процессов, *P1* … *Pn*.
- Пусть *m* — число **классов** ресурсов,
  E — это вектор существующих ресурсов
  Ei — количество ресурсов класса *i* (где $1 \le i \le m$).
- Пусть *A* - вектор доступных ресурсов,
  Ai – количество экземпляров ресурса *i*, доступных на данный момент.

ИНСТИТУТ МАТЕМАТИКИ МЕХАНИКИ КОМПЬЮТЕРНЫХ НАУК имени И.И. Воровича

Обнаружение взаимоблокировки при использовании нескольких ресурсов каждого типа

- *C* — матрица текущего распределения ресурсов.
- *R* — матрица запросов к ресурсам.
- *Cij* — количество экземпляров ресурса *j* (*Cij* ≤ *Ej*), которое удерживается процессом *Pi*.
- *Rij* — количество экземпляров ресурса *j* (*Rij* ≤ *Ej*), которое хочет получить процесс *Pi*.

ИНСТИТУТ МАТЕМАТИКИ МЕХАНИКИ КОМПЬЮТЕРНЫХ НАУК имени И.И. Воровича

Обнаружение взаимоблокировки при использовании нескольких ресурсов каждого типа

Существующие ресурсы

$(E_1, E_2, E_3, \ldots, E_m)$

Доступные ресурсы

$(A_1, A_2, A_3, \ldots, A_m)$

Матрица текущего распределения

$$\begin{bmatrix} C_{11} & C_{12} & C_{13} & \cdots & C_{1m} \\ C_{21} & C_{22} & C_{23} & \cdots & C_{2m} \\ \vdots & \vdots & \vdots & & \vdots \\ C_{n1} & C_{n2} & C_{n3} & \cdots & C_{nm} \end{bmatrix}$$

Строка n в данный момент предоставлена процессу n

Матрица запроса

$$\begin{bmatrix} R_{11} & R_{12} & R_{13} & \cdots & R_{1m} \\ R_{21} & R_{22} & R_{23} & \cdots & R_{2m} \\ \vdots & \vdots & \vdots & & \vdots \\ R_{n1} & R_{n2} & R_{n3} & \cdots & R_{nm} \end{bmatrix}$$

Строка 2 нужна процессу 2

ИНСТИТУТ МАТЕМАТИКИ МЕХАНИКИ КОМПЬЮТЕРНЫХ НАУК имени И.И. Воровича

Обнаружение взаимоблокировки при использовании нескольких ресурсов каждого типа

- Для всех j (где $1 \le j \le m$) $$\sum_{i=1}^{n} C_{ij} + A_j = E_j.$$
- Алгоритм обнаружения взаимоблокировок основан на сравнении векторов.
- Введем *соотношение "≤"* для векторов так: $A \le B$ тогда и только тогда, когда $Ai \le Bi$ для всех $1 \le i \le m$.

ИНСТИТУТ МАТЕМАТИКИ МЕХАНИКИ КОМПЬЮТЕРНЫХ НАУК имени И.И. Воровича

Алгоритм

1. Объявляем все процессы непомеченными.
2. Ищем непомеченный процесс *Pi*, для которого i-я строка матрицы *R* меньше или равна *A*.
3. Если такой процесс найден, добавляем к *A* значения *i-й* строки матрицы *C*, ставим метку на процесс *Pi* и идём к шагу 2.
4. Если такого процесса нет, алгоритм завершает работу.

- По окончании работы алгоритма все непомеченные процессы, если таковые имеются, считаются участвующими во взаимоблокировке

ИНСТИТУТ МАТЕМАТИКИ МЕХАНИКИ КОМПЬЮТЕРНЫХ НАУК имени И.И. Воровича

Проверка наличия тупика

Накопители на магнитной ленте, Плоттеры, Сканеры, Компакт-диски

$E = (4 \quad 2 \quad 3 \quad 1)$

Накопители на магнитной ленте, Плоттеры, Сканеры, Компакт-диски

$A = (2 \quad 1 \quad 0 \quad 0)$

Матрица текущего распределения

$$C = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 2 & 0 & 0 & 1 \\ 0 & 1 & 2 & 0 \end{bmatrix}$$

Матрица запросов

$$R = \begin{bmatrix} 2 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 2 & 1 & 0 & 0 \end{bmatrix}$$

ИНСТИТУТ МАТЕМАТИКИ МЕХАНИКИ КОМПЬЮТЕРНЫХ НАУК имени И.И. Воровича

Проверка наличия тупика

Накопители на магнитной ленте, Плоттеры, Сканеры, Компакт-диски

E = (4 2 3 1)

Накопители на магнитной ленте, Плоттеры, Сканеры, Компакт-диски

A = (2 1 0 0)

A = (2 2 2 0)

Матрица текущего распределения

$$C = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 2 & 0 & 0 & 1 \\ 0 & 1 & 2 & 0 \end{bmatrix}$$

Матрица запросов

$$R = \begin{bmatrix} 2 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 2 & 1 & 0 & 0 \end{bmatrix}$$

ИНСТИТУТ МАТЕМАТИКИ МЕХАНИКИ КОМПЬЮТЕРНЫХ НАУК имени И.И. Воровича

Проверка наличия тупика

Накопители на магнитной ленте | Плоттеры | Сканеры | Компакт-диски

E = (4 2 3 1)

Накопители на магнитной ленте | Плоттеры | Сканеры | Компакт-диски

A = (2 1 0 0)

A = (2 2 2 0)

A = (4 2 2 1)

Матрица текущего распределения

$$C = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 2 & 0 & 0 & 1 \\ 0 & 1 & 2 & 0 \end{bmatrix}$$

Матрица запросов

$$R = \begin{bmatrix} 2 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 2 & 1 & 0 & 0 \end{bmatrix}$$

ИНСТИТУТ МАТЕМАТИКИ МЕХАНИКИ КОМПЬЮТЕРНЫХ НАУК имени И.И. Воровича

Проверка наличия тупика

Накопители на магнитной ленте | Плоттеры | Сканеры | Компакт-диски

E = (4 2 3 1)

Накопители на магнитной ленте | Плоттеры | Сканеры | Компакт-диски

A = (2 1 0 0)

A = (2 2 2 0)

A = (4 2 2 1)

A = (4 2 3 1)

Матрица текущего распределения

$$C = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 2 & 0 & 0 & 1 \\ 0 & 1 & 2 & 0 \end{bmatrix}$$

Матрица запросов

$$R = \begin{bmatrix} 2 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 2 & 1 & 0 & 0 \end{bmatrix}$$

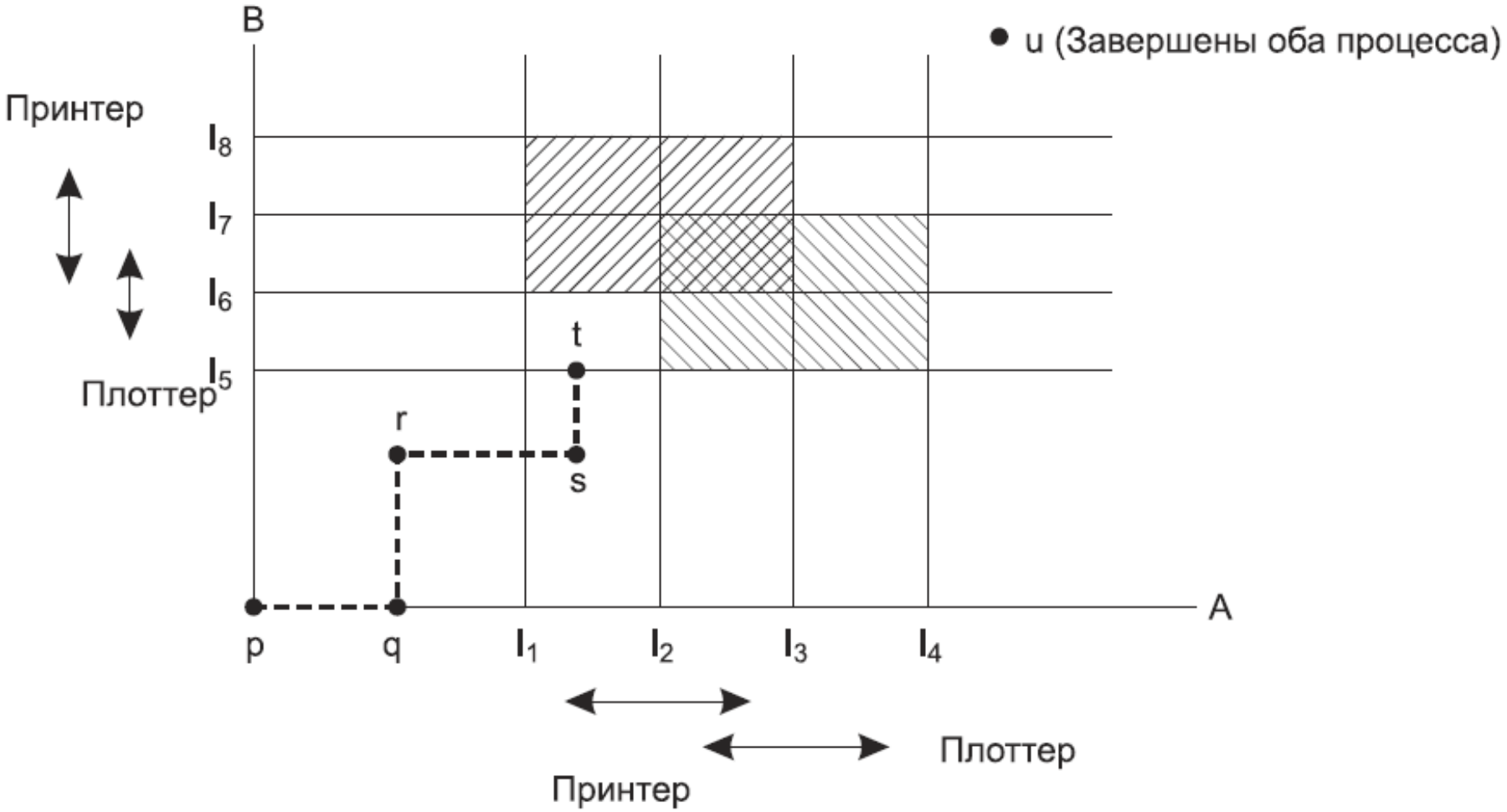
ИНСТИТУТ МАТЕМАТИКИ МЕХАНИКИ КОМПЬЮТЕРНЫХ НАУК имени И.И. Воровича

Выход из взаимоблокировки

- Восстановление за счет приоритетного овладения ресурсом
  - вмешательство администратора
- Восстановление путем отката
  - контрольные точки процесса
- Восстановление путем уничтожения процессов
  - вмешательство администратора

ИНСТИТУТ МАТЕМАТИКИ МЕХАНИКИ КОМПЬЮТЕРНЫХ НАУК имени И.И. Воровича

Уклонение от взаимоблокировок



ИНСТИТУТ МАТЕМАТИКИ МЕХАНИКИ КОМПЬЮТЕРНЫХ НАУК имени И.И. Воровича

Безопасное состояние

- Состояние считается **безопасным,** если существует какой-то порядок планирования, при котором каждый процесс может доработать до конца, даже если все процессы внезапно и срочно запросят максимальное количество ресурсов.
- Пусть есть 10 экземпляров ресурса

	Имеет	Max
A	3	9
B	2	4
C	2	7

Свободно: 3

	Имеет	Max
A	3	9
B	4	4
C	2	7

Свободно: 1

	Имеет	Max
A	3	9
B	—	—
C	2	7

Свободно: 5

	Имеет	Max
A	3	9
B	—	—
C	7	7

Свободно: 0

	Имеет	Max
A	3	9
B	—	—
C	—	—

Свободно: 7

Небезопасное состояние

	Имеет	Max
A	3	9
B	2	4
C	2	7

Свободно: 3

	Имеет	Max
A	4	9
B	2	4
C	2	7

Свободно: 2

	Имеет	Max
A	4	9
B	4	4
C	2	7

Свободно: 0

	Имеет	Max
A	4	9
B	—	—
C	2	7

Свободно: 4

- В безопасном состоянии система может гарантировать, что все процессы закончат свою работу, а в небезопасном состоянии такой гарантии нет.

ИНСТИТУТ МАТЕМАТИКИ МЕХАНИКИ КОМПЬЮТЕРНЫХ НАУК имени И.И. Воровича

Алгоритм банкира

- Позволяет избегать взаимоблокировки
- Разработан Дейкстрой (Dijkstra, 1965)

	Имеет	Max
A	0	6
B	0	5
C	0	4
D	0	7

Свободно: 10

	Имеет	Max
A	1	6
B	1	5
C	2	4
D	4	7

Свободно: 2

	Имеет	Max
A	1	6
B	2	5
C	2	4
D	4	7

Свободно: 1

ИНСТИТУТ МАТЕМАТИКИ МЕХАНИКИ КОМПЬЮТЕРНЫХ НАУК имени И.И. Воровича

Алгоритм банкира

- Каждый запрос по мере поступления проверяется. Приведет ли его удовлетворение к безопасному состоянию?
- Если да, то запрос удовлетворяется, в противном случае запрос откладывается (до лучших времен).

ИНСТИТУТ МАТЕМАТИКИ МЕХАНИКИ КОМПЬЮТЕРНЫХ НАУК имени И.И. Воровича

Алгоритм банкира для нескольких типов ресурсов

Процесс	Накопители на магнитных дисках	Плоттеры	Сканеры	Устройства для чтения компакт-дисков
A	3	0	1	1
B	0	1	0	0
C	1	1	1	0
D	1	1	0	1
E	0	0	0	0

Распределенные ресурсы

Процесс	Накопители на магнитных дисках	Плоттеры	Сканеры	Устройства для чтения компакт-дисков
A	1	1	0	0
B	0	1	1	2
C	3	1	0	0
D	0	0	1	0
E	2	1	1	0

Ресурсы, которые еще нужны

E = (6342)
P = (5322)
A = (1020)

ИНСТИТУТ МАТЕМАТИКИ МЕХАНИКИ КОМПЬЮТЕРНЫХ НАУК имени И.И. Воровича

Алгоритм банкира для нескольких типов ресурсов

- Ищем в матрице *R* строку, соответствующую процессу, чьи неудовлетворенные потребности в ресурсах меньше или равны вектору *A*. Если такой строки не существует, то система в конце концов войдет в состояние взаимоблокировки.
- Допускаем, что процесс, чья строка была выбрана, запрашивает все необходимые ему ресурсы (возможность чего гарантируется) и завершает свою работу. Отмечаем этот процесс как завершенный и прибавляем все его ресурсы к вектору *A*.
- Повторяем шаги 1 и 2 до тех пор, пока либо все процессы будут помечены как завершенные (в этом случае исходное состояние было безопасным), либо не останется процессов, чьи запросы могут быть удовлетворены (в этом случае система не была в безопасном состоянии).

ИНСТИТУТ МАТЕМАТИКИ МЕХАНИКИ КОМПЬЮТЕРНЫХ НАУК имени И.И. Воровича

Предотвращение взаимоблокировки

1. Условие взаимного исключения.
2. Условие удержания и ожидания.
3. Условие невыгружаемости.
4. Условие циклического ожидания.

Условие взаимного исключения

- данные - сделать доступными только для чтения
- принтер - использование очереди на печать (спулинга).
  А если данные не готовы?

- Избегать выделения ресурса, если в нем нет насущной потребности. Стараться, чтобы как можно меньше процессов могло фактически требовать получения ресурса

Условие удержания и ожидания

- запрашивать ресурсы до начала работы
- недостатки
  - неоптимальное использование ресурсов
  - слабовыполнимо

- Можно потребовать от процесса, запрашивающего ресурс, вначале временно высвободить все ресурсы, удерживаемые им на данный момент.

ИНСТИТУТ МАТЕМАТИКИ МЕХАНИКИ КОМПЬЮТЕРНЫХ НАУК имени И.И. Воровича

Условие невыгружаемости

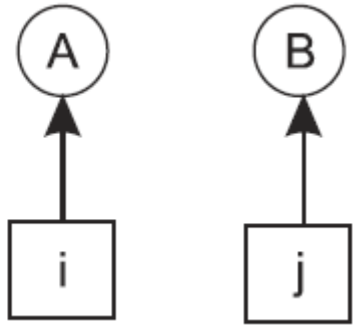
- виртуализация ресурсов
- недостатки
  - не везде применимо. Записи в БД или в таблицах ОС, при использовании должны быть заблокированы , и здесь закладывается потенциальная вероятность взаимоблокировки.

ИНСТИТУТ МАТЕМАТИКИ МЕХАНИКИ КОМПЬЮТЕРНЫХ НАУК имени И.И. Воровича

Условие циклического ожидания

- процессу в любой момент времени дано право только на один ресурс!
- недостатки
  - не везде применимо
- единая нумерация ресурсов и все запросы в порядке нумерации ресурсов

1. Принтер
2. Сканер
3. Плоттер
4. Накопитель на магнитной ленте
5. Устройство для чтения компакт-дисков



Livelock

- В отличие от deadlock система не «застревает», а занимается бесполезной работой. Её состояние постоянно меняется — но, тем не менее, она «зациклилась», не производит никакой полезной работы.

ИНСТИТУТ МАТЕМАТИКИ МЕХАНИКИ КОМПЬЮТЕРНЫХ НАУК имени И.И. Воровича

Зависание ("голодание")

- Политика предоставления ресурсов, приводящая к тому, что некоторые процессы никогда не будут обслужены, даже если они не находятся в состоянии взаимоблокировки.

- Зависания можно избежать за счет использования политики распределения ресурсов «первым пришел — первым и обслужен»

ИНСТИТУТ МАТЕМАТИКИ МЕХАНИКИ КОМПЬЮТЕРНЫХ НАУК имени И.И. Воровича

Домашнее задание

- Читать книгу Таненбаум Э., Бос Х. Современные операционные системы, стр. 488-520.