# Операционные системы Лекция 7

## Виртуальная память

### ?Задача коллективного использования?

Программы, которые коллективно используются должны быть реентерабельными, то есть допускать повторную входимость.

В результате такой потребности (это в особенности касается данных), такую страницу надо отдельно обозначать для каждого процесса, при этом устанавливать уровень доступа к ней. Проблема: теряется доступ к конкретному набору данных.

## Алгоритмы замещения страниц (Page replacement)

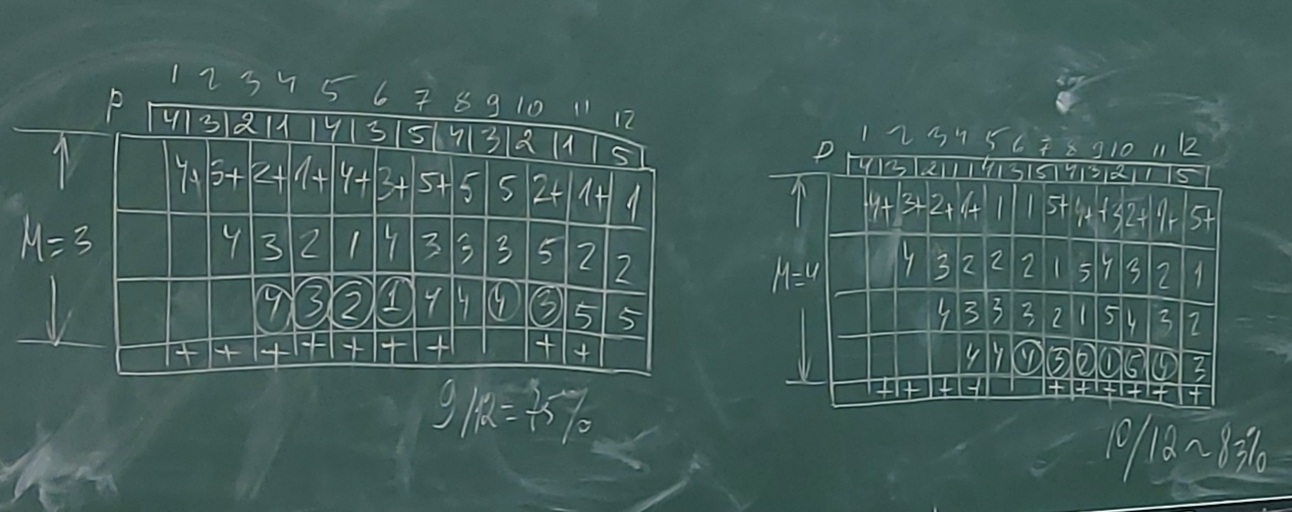
**Способы (Модели):**

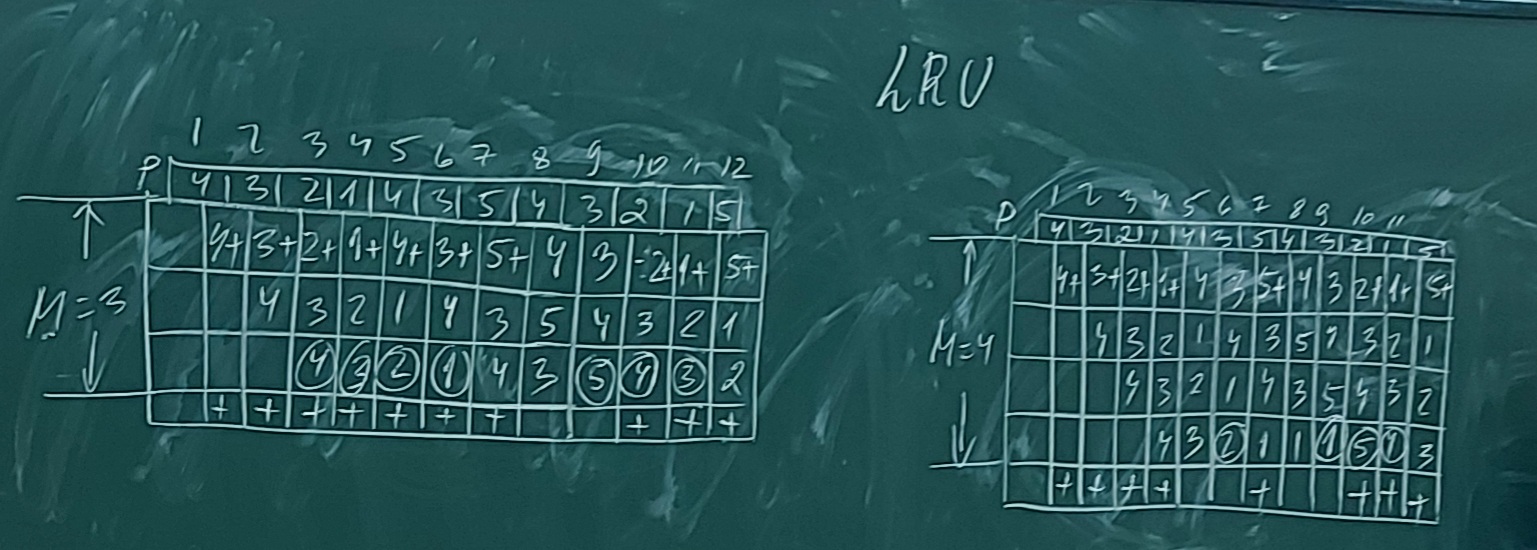
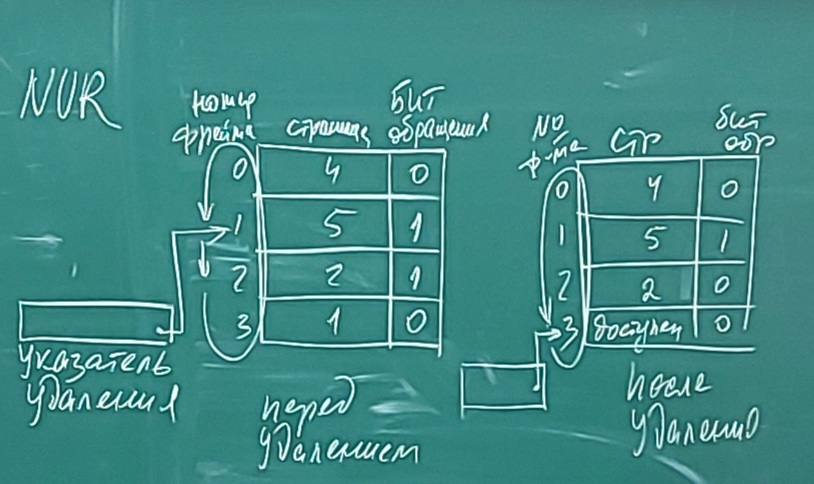
1. **Выталкивание случайной страницы** (первая попавшееся).  
   Выталкиваем во вторичную память.   
   Данный алгоритм хар-я малыми издержками, не явл.   
   Недостатки:
   1. Может быть вытолкнута часто используемая или только что загруженная страница.

Преимущества:

* 1. Малые накладные расходы

1. **FIFO.** Выталкивается та, которая дольше всего находится в памяти.  
   Каждой странице присваивается либо временная метка либо…  
   Для реализации этого способа нужно организовать очередь либо хранить время.  
   Недостаток:
   1. Может быть выгружена часто используемая страница.

**Аномалия FIFO**: (особенность) Существуют такие траектории загрузки страниц, когда увеличение страничной памяти ведет к увеличению страничных прерываний.  
рисунок (см. тетрадь):

1. **LRU**LRU полностью соответствует свойству локальности программ.  
   Свойство локальности утверждает – если обращение было к странице, то наиболее вероятно, что следующие обращения будут к этой же странице.  
   Это свойство следует из тех особенностей программ, которые мы с вами пишем. Во-первых в наших программах (если отбросить go to), существует только три типа действия: следование, ветвление и повторение.  
   (Любой алгоритм может быть представлен этими тремя структурами (следование, ветвление и повторение))  
   Следовательно из свойства локальности => наших программах наиболее вероятно, что команды следуют одна за другой.  
   Кроме этого, если посмотреть на эти рисунки, то первые 3 строки в обоих таблицах одинаковы. Это **свойство включения**.  
   Оно означает если какая-то страница выбрана в реализации (p, M, t), то это же страница будет выбрана в этой же реализации (p, M+1, t)  
   Алгоритм LRU относится к… которые называются стековыми алгоритмами.  
   Но в рассматриваемом виде он не используется, т.к. он крайне затратный.  
   Для его реализации необходимо или модифицировать временную метку при каждом обращении и … связный список… перемещая… в хвост…  
   При этом, как часто возможно обращение к странице? (Ответ - ОЧЕНЬ ОЧЕНЬ ЧАСТО – **на каждой команде или** еще чаще)
2. **LFU (**Least Frequency Used**) -** Наименее часто используемая страница  
   Здесь используется счётчик обращений к странице. Этот счётчик инкрементируется при каждом ообращении к странице.  
   Недостаток: Может быть вытеснена только что загруженная страница, не набравшая число обращений.
3. **NUR** (Not Used Recently) страница не используемая в последнее время. Этот алгоритм является аппроксимацией LRU.  
   Для его реализации у каждой страницы есть бит обращения.  
   Этот бит обращения, периодически сбрасывается в ноль. Соответственно при обращении к странице, он устанавливается в 1.  
   => Вытесняем только из тех у которых бит сброшен. И это показывает, что с момента последнего сброса, обращения к этой странице, не было.  
   Для реализации (также) вводится указатель удаления.  
   На первом рисунке - загружается 5ая страница. Если в этот момент необходимо удалить страницу, то проверку значений битов обращения, будет выполнятся со второго кадра (фрэйма – поэтому показана цикличность).  
   Кроме бита обращения, вводится бит модификации (Dirty - грязный). Делается это для … (не нужно копирование для не модифицируемых страниц т.к. точная копия лежит в памяти – исключается копирование).  
   Возможны 4е ситуации: 00, 01, 10, 11. 01 – модификация была до сброса бита обращения в ноль.

## Следующий заголовок

Фирма IBM очень тщательно проводила исследования страничного преобразования.

## Поведение программ и производительность

Еще на прошлой лекции стало ясно, как затратен … управления виртуальной памятью. Но есть важный плюс – **увеличение мультизадачности** или увеличение уровня мультипрограммирования (за счёт того, что снимаются ограничения, которые накладываются размером, физической памяти). Но цена – это загруженность процессора соответствующими (обработками) действиями.

Давайте рассмотрим график зависимости процента страниц, к которым обращается типичный процесс от времени. От начала процесса до его завершения.

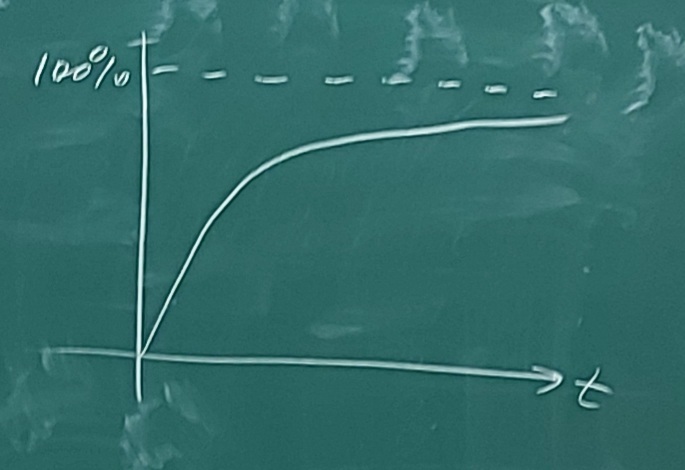


График отображает, что процесс в какие-то отрезки времени не обращается ко всем своим страницам.

График НЕ ЛИНЕЙНЫЙ! (это важный момент) – то есть этот изгиб (до него график линейный, а после не линейный), сначала интенсивно подгружает страницы, а потом прекращает делать это интенсивно.

## Теория рабочего множества

Зависимость страничных прерываний от объёма памяти является обобщённой мерой страничного поведения программы. При этом как видно из фотографии памяти (давали на лекции), поведение программы во времени не является стабильным и равномерным. То есть за время своего выполнения программа обращается в течении некоторого интервала времени к разному количеству страниц.

Денинг в 1986 г. предложил в качестве локальной меры производительности взять число страниц, к которым программа обращается за некоторый интервал дельтаT W – working set. .

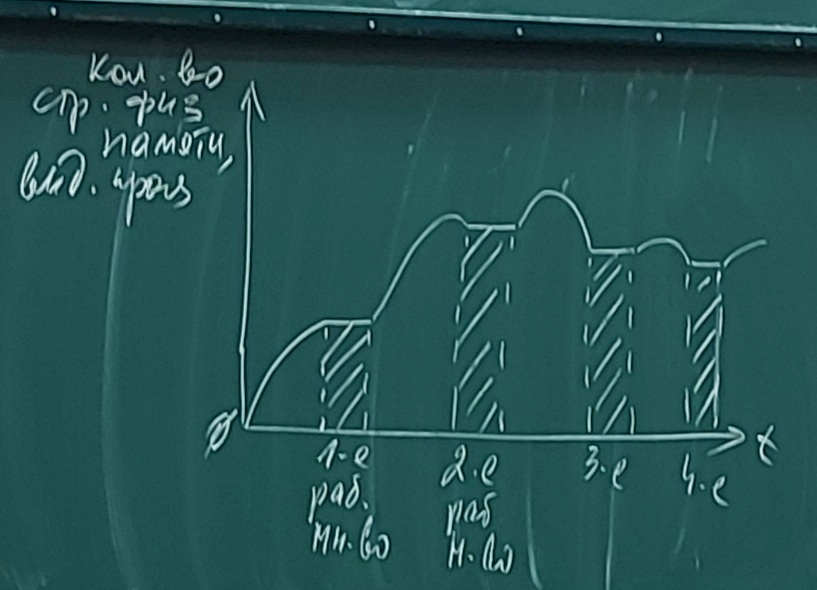
Размер рабочего набора (W) – является … То есть при увеличении dT, число страниц будет стремится к некоторому пределу L, который определяет кол-во страниц необходимых процессу для эффективного выполнения. Под эффективным выполнением понимается выполнение процесса без страничных прерываний.

Другими словами: если процессу получается загрузить в память своё рабочее множество, то есть все страницы, к которым он обращается в течении некоторого интервала времени dT, то процесс будет выполняться без страничных прерываний.

Модно перефразировать: для того, чтобы процесс выполнялся эффективно, необходимо чтобы он мог загрузить в память всё своё рабочее множество.

А это уже оценка объёма памяти необходимого для эффективного выполнения.

Предсказать какие страницы, когда будут запрошены НЕВОЗМОЖНО. Но по факту, процесс должен иметь возможность, загрузить в память всё своё рабочее множество. Если ему это не удастся, то возникнет интенсивная подкачка (Trashing). То есть подкачка одних и тех же страниц.

**Картинка, которая ярко иллюстрирует данную мысль.**

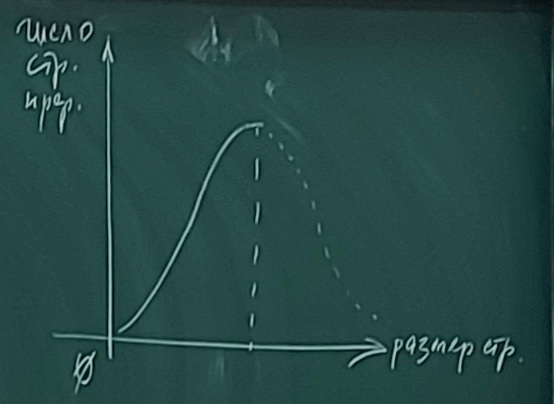
Процесс интенсивно подгружает страницы пока не возникнет первое рабочее множество. Если загрузить все страницы, то пока что не будет страничных прерываний.

Используется также название Hit rate – это число удачных обращений (страничных удач).

Важный вопрос – выбор размера страницы. На него влияют два соображения:

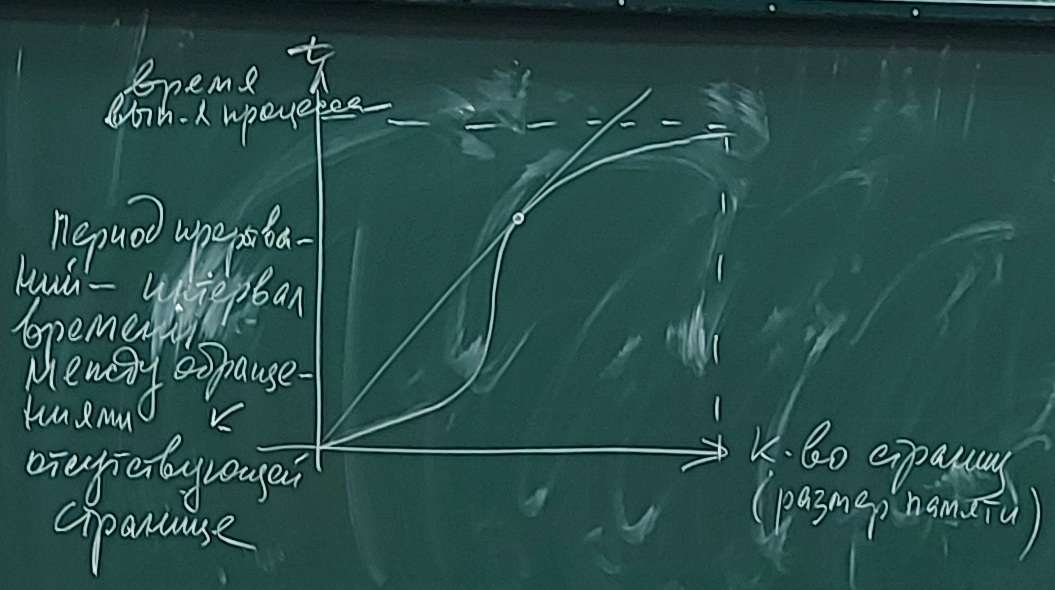
1. На маленькой странице выполняется больший процент команд.
2. Чем меньше страница, тем больше таблица страниц (каждую страницу надо описать соответствующим дескриптором)

Ну и есть график показывающий влияние размера страницы при фиксированном объёме памяти на число страничных прерываний.

**Рисунок**

Из него видно, что кол-во прерываний растёт с увеличением размера странице. Чем больше страницы, тем меньше процент выполнения команд на ней и потребуется переход на другую страницу. А потом идёт спад, потому что размер страницы стал соразмерим с размером программы.

Еще один график: график зависимости длительности периода между прерываниями или так называемая кривая времени жизни.

**Рисунок**

Вот этот перегиб происходит из-за того, что в некоторые моменты времени в памяти оказывается всё рабочее множество страниц процесса.

?Интервал между прерываниями называется временем жизни.?

Всё это косвенно подтверждает факт существования рабочего множества.

Контроль кол-ва прерываний процесса является важнейшим показателем правильного выделения памяти. Под правильным выделением можно понимать необходимость предоставления процессу нужного ему кол-ва страниц.

## Глобальное и локальное замещение страниц

Под глобальным замещением понимается выгрузка любой страницы любого процесса для того, чтобы процесс мог загрузить свою очередную страницу.

Под локальным замещением понимается выгрузка только страниц данного процесса.

В UNIX демон страниц. А в виндах ?swiper?

## Управление памятью сегментами по запросу

Страница является единицей физического деления памяти. Её размер устанавливается системой. (У страницы размер не обязательно 4Кб. Просто такой размер оптимален по количеству страничных прерываний.)

Сегмент является единицей логического деления памяти. Поэтому дополнительно о сегменте должен быть известен его размер (размер сегмента). (Его размер определяется объемом кода. Любой сегмент создается такого размера, который требуется программе.)

Схема преобразования (? виртуального адреса?) при этом будет выглядеть следующим образом:

(В следующий раз)