Виртуализация на паметта

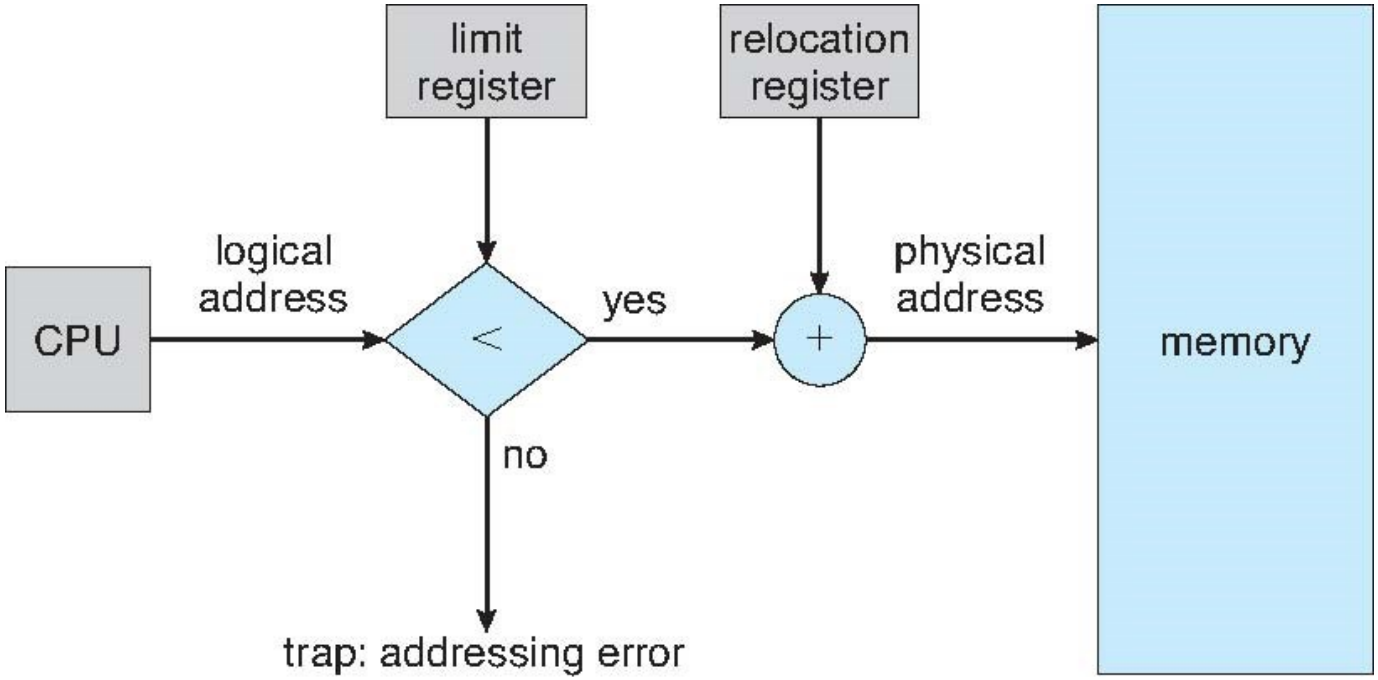
проф. д-р инж. Христо Вълчанов

http://cs.tu-varna.bg

Непрекъсната организация на паметта

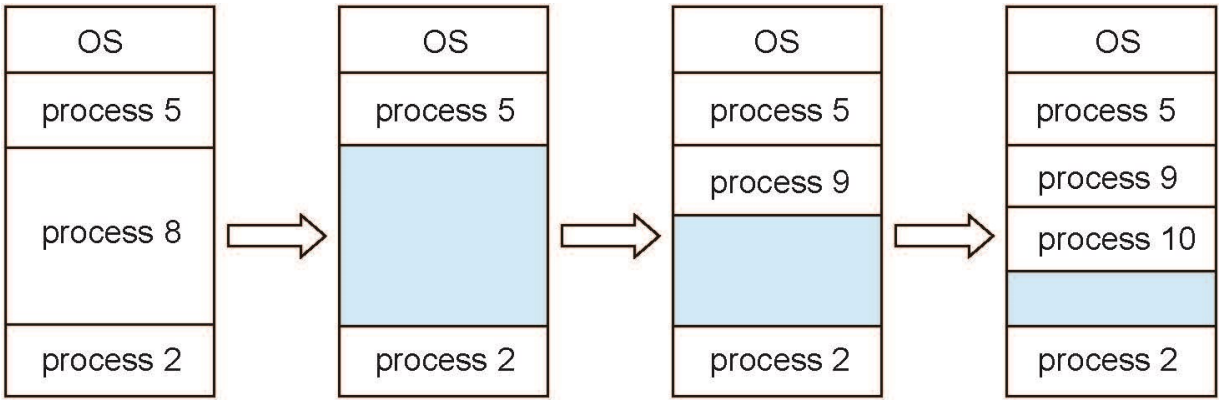
- ОП се разделя обикновено на два дяла:
  - Резидентна ОС, типично заемаща паметта в началото на адресното пространство
  - Потребителските процеси заемат горната част на паметта
  - Всеки процес се съдържа в единична непрекъсната секция от паметта
- Използват се регистри на преместване:
  - Базов регистър за начален адрес на разполагане;
  - Ограничителен регистър – всеки логически адрес трябва да бъде по-малък от стойността на този регистър

Непрекъсната организация на паметта



Дялове на паметта

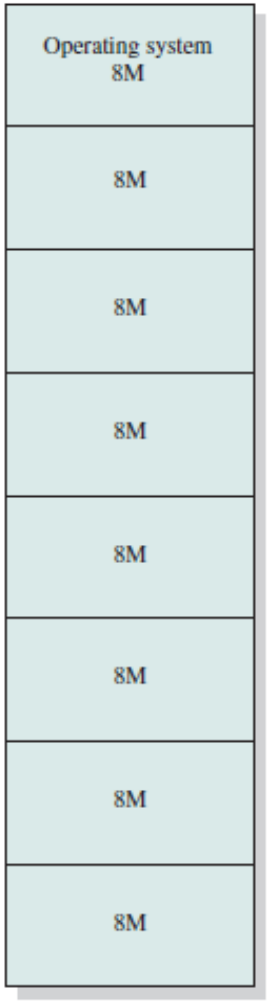
- Паметта за потребителските процеси се разделя на дялове.
- Всеки дял съдържа един процес.
- При завършване на процес, дялът се освобождава за друг процес.
- Възможност за получаване на фрагментиране



Фрагментиране на паметта

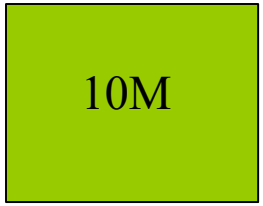
- **Външно** – общият обем памет е достатъчен за процес, но пространството не е непрекъснато и не може да се използва.
- **Вътрешно** – заетата памет в дяла е по-малка от неговия размер.

Дялове с фиксиран размер

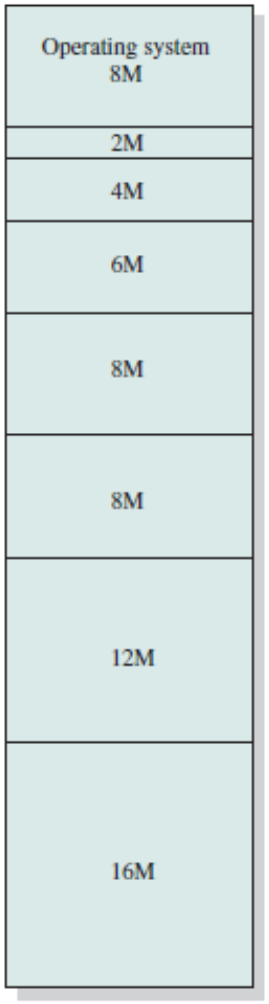


- Всички са с еднакъв размер

Нов процес

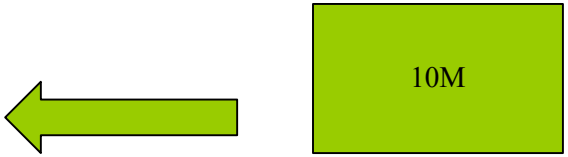


Дялове с фиксиран размер

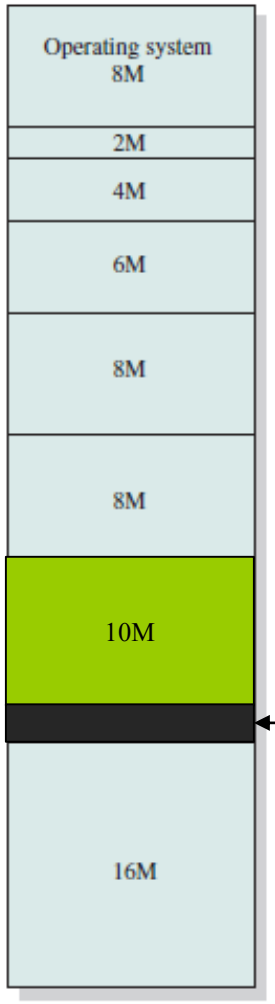


- Дяловете са с различен размер

Нов процес

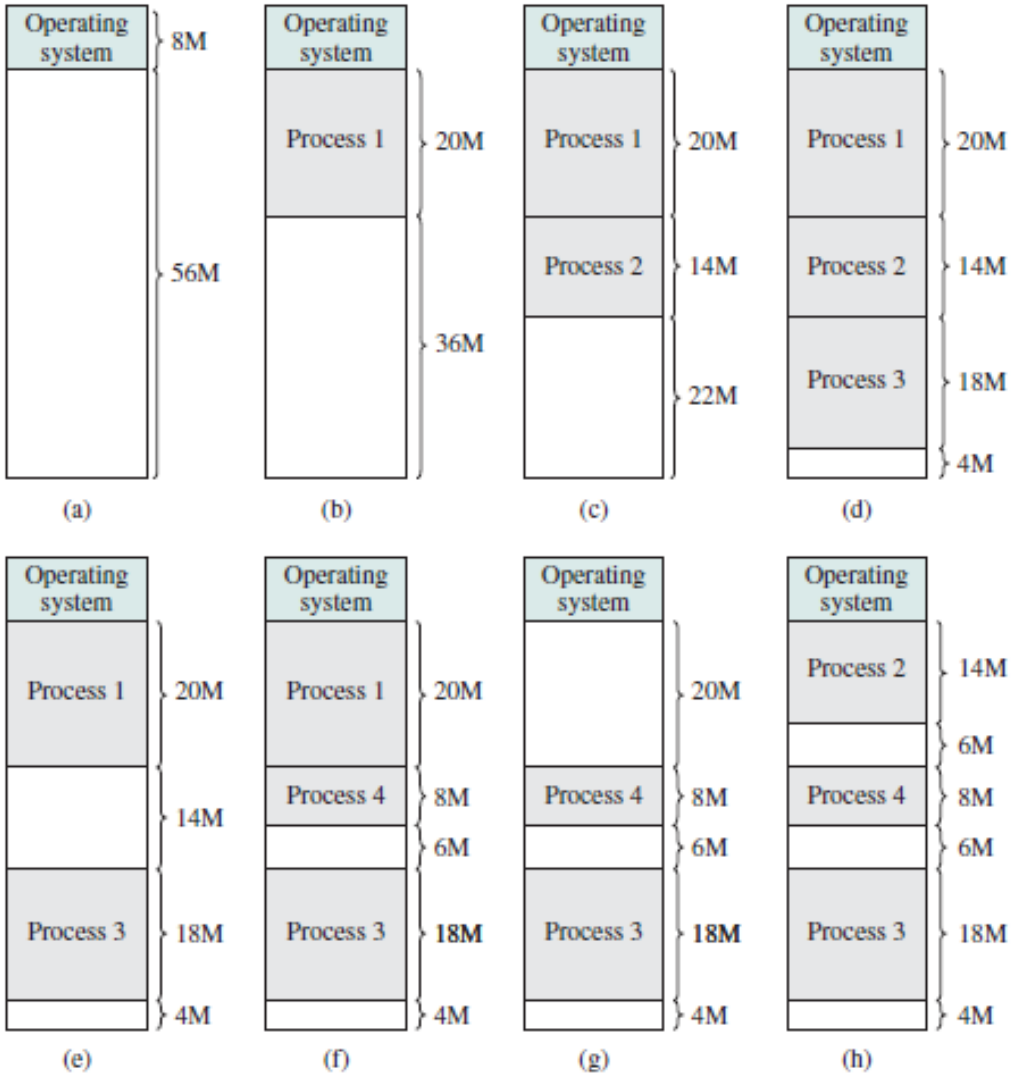


Вътрешна фрагментация

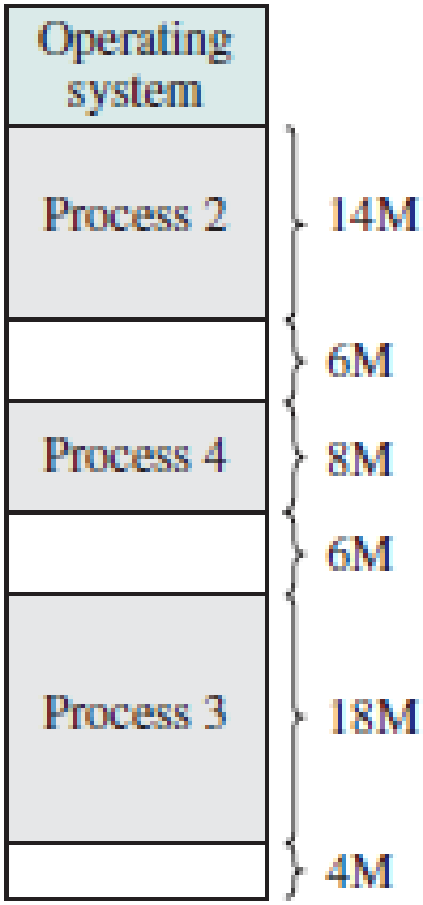


Вътрешно фрагментиране в дяла

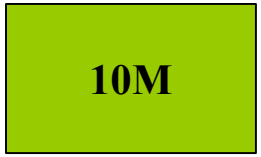
Дялове с динамичен размер



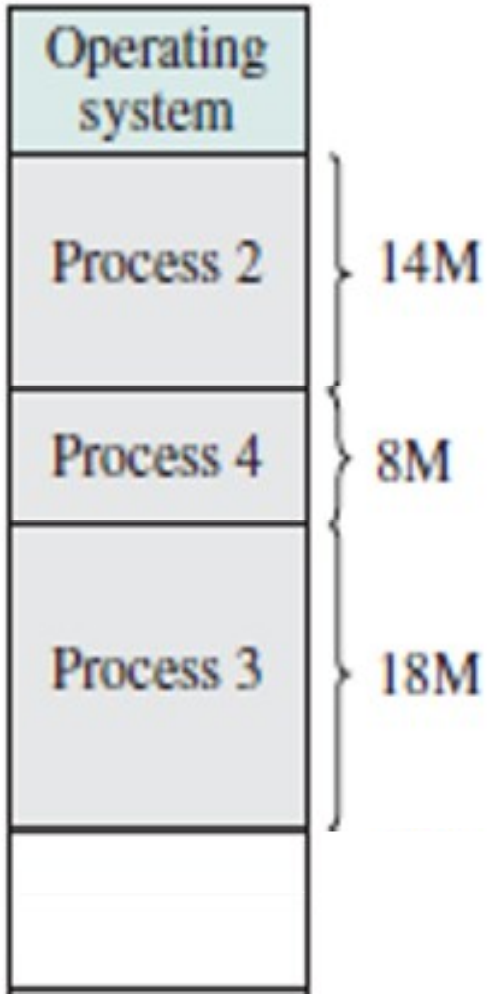
Външна фрагментация



Нов процес



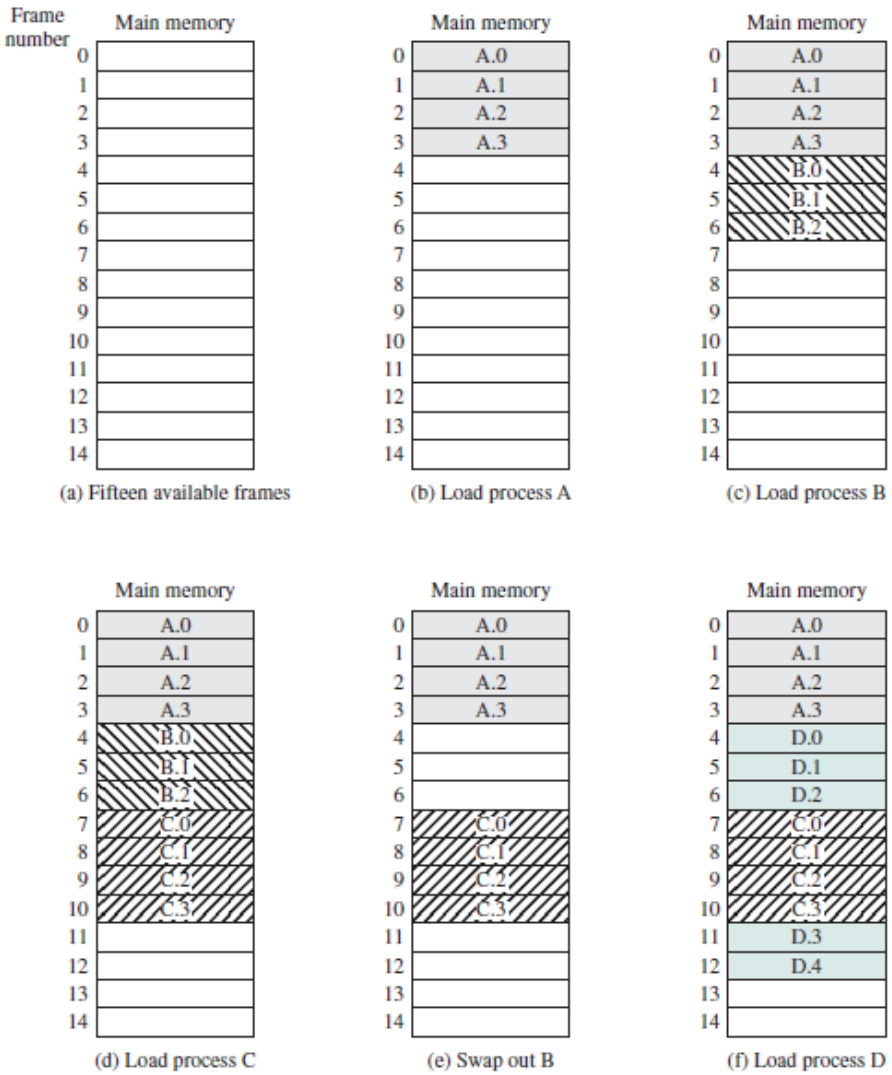
Дефрагментиране на паметта



Странична организация на ОП

- Физическата памет се разделя на блокове с фиксиран размер – фреймове от 512 – 16Mb. Размерът на фрейма е степен на 2.
- Логическата памет се разделя на блокове със същия размер – страници.
- Използва се таблица на страниците за транслиране на логическия адрес.
- Елиминира се външната фрагментация.
- Възможна е вътрешна фрагментация.

Зареждане на процеси



Таблици на процесите

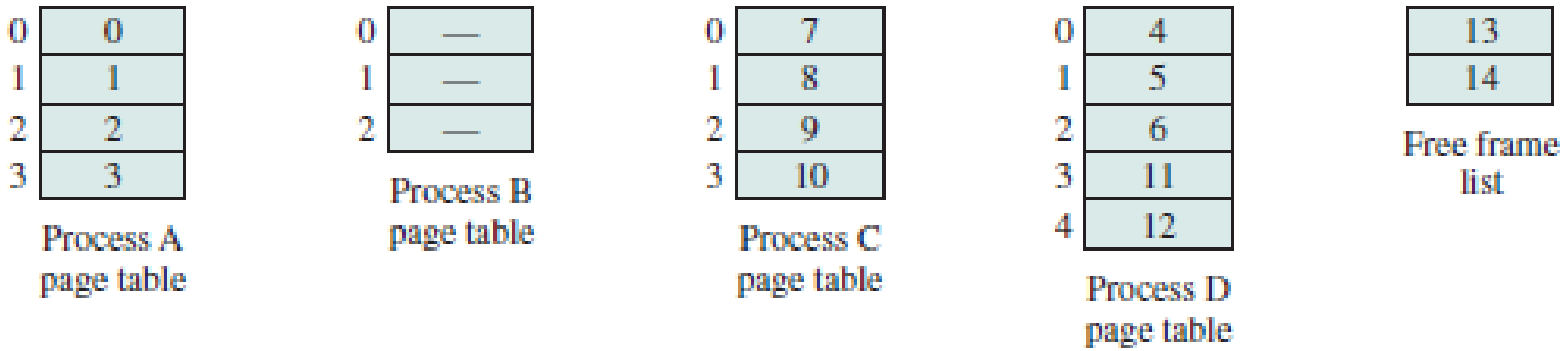
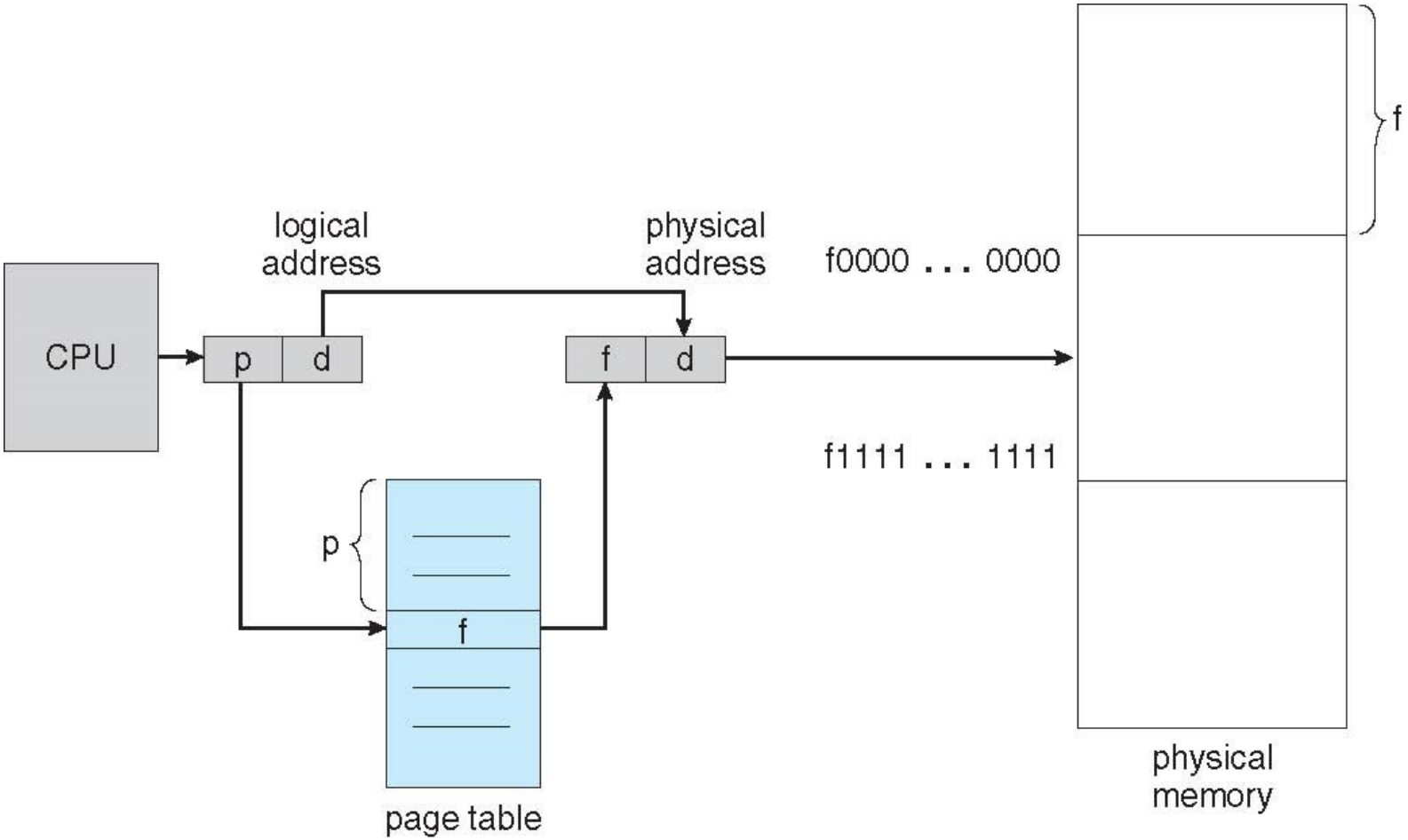


Схема на транслиране

- Логическото адресно пространство е 2^m
- Размерът на страница е 2^n

page number	page offset
p	d
m -n	n

Схема на транслиране

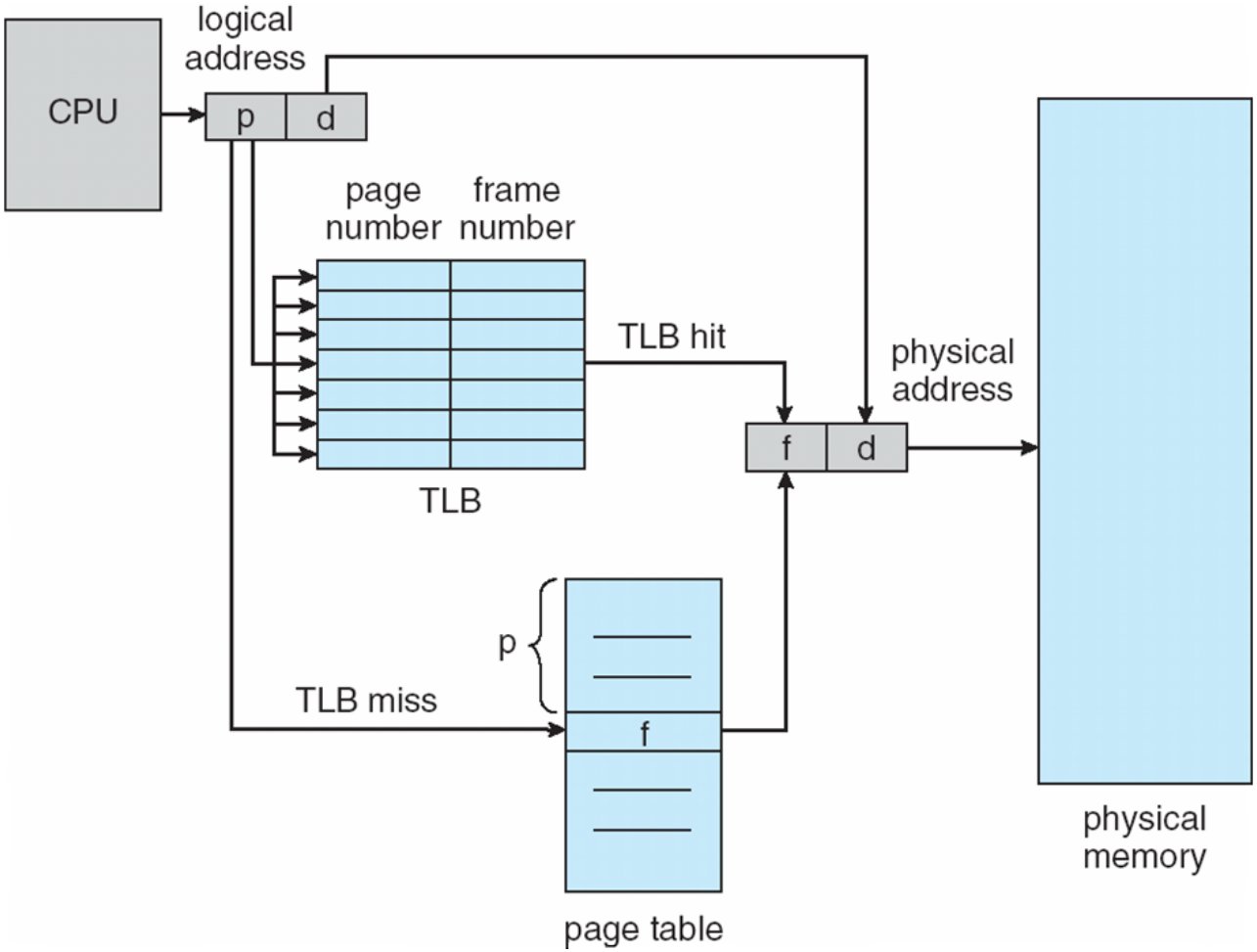


Асоциативна памет

- Ако таблицата на страниците е в ОП е необходим два пъти достъп до паметта – един за таблицата и един за инструкцията/данните.
- Използва се бърза кеш памет – Translation Look-aside Buffer (TLB).
- Асоциативна памет – паралелно търсене

Page #	Frame #

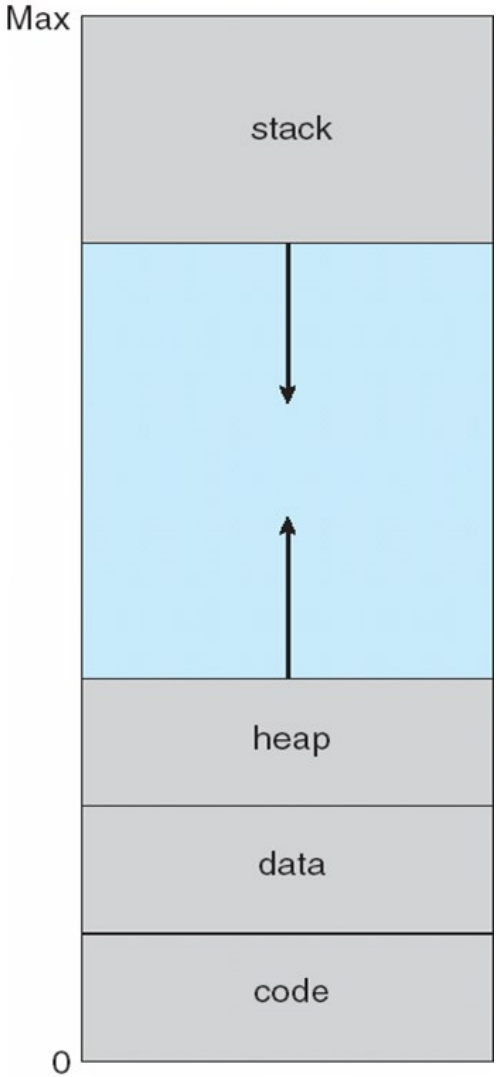
TLB транслиране



Виртуална памет

- Разделяне на логическата от физическата памет.
- Позволява използване на повече виртуална памет от реално наличната физическа.
- Позволява наличие в паметта само на част от програмата, която се изпълнява.
- Позволява различни процеси да си споделят файлове и памет.
- Реализира се основно на базата на механизма на заявки за страници (*demand paging*).

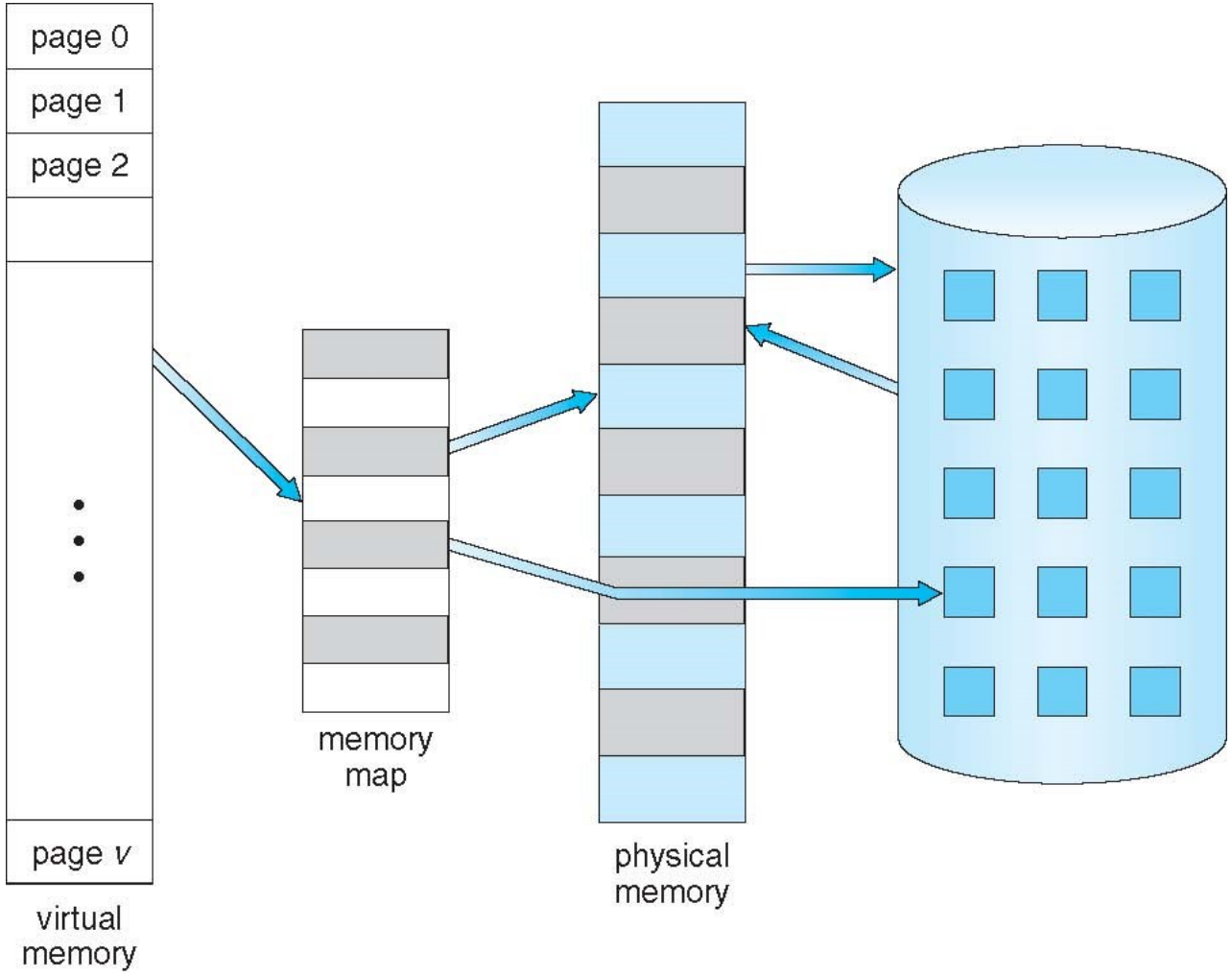
Виртуално адресно пространство



Прехвърляне на процеси (swapping)

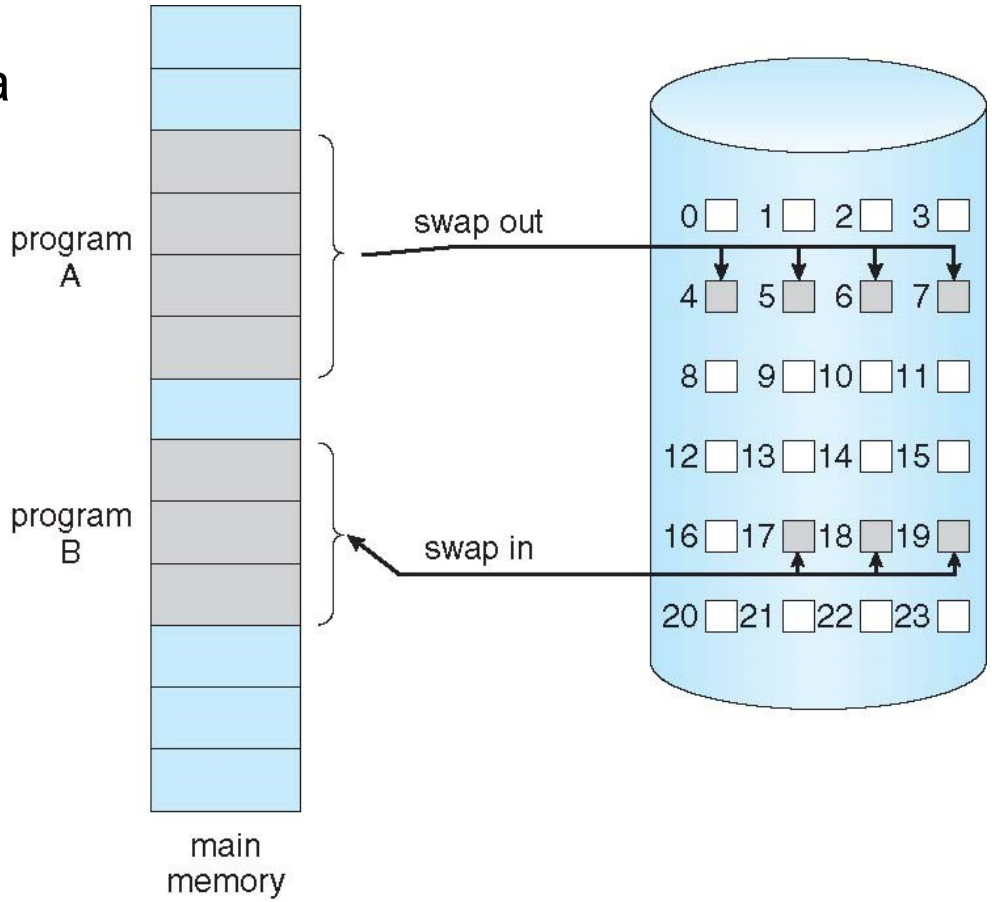
- Съхраняване на паметта за процес на външен носител или зареждането й от диска в ОП.
- Действието се стартира при изчерпване на наличната физическа памет.
- Може да се прехвърля част от процеса.
- Типично е свързано с концепцията за виртуална памет.

Виртуална памет



Заявки за страници

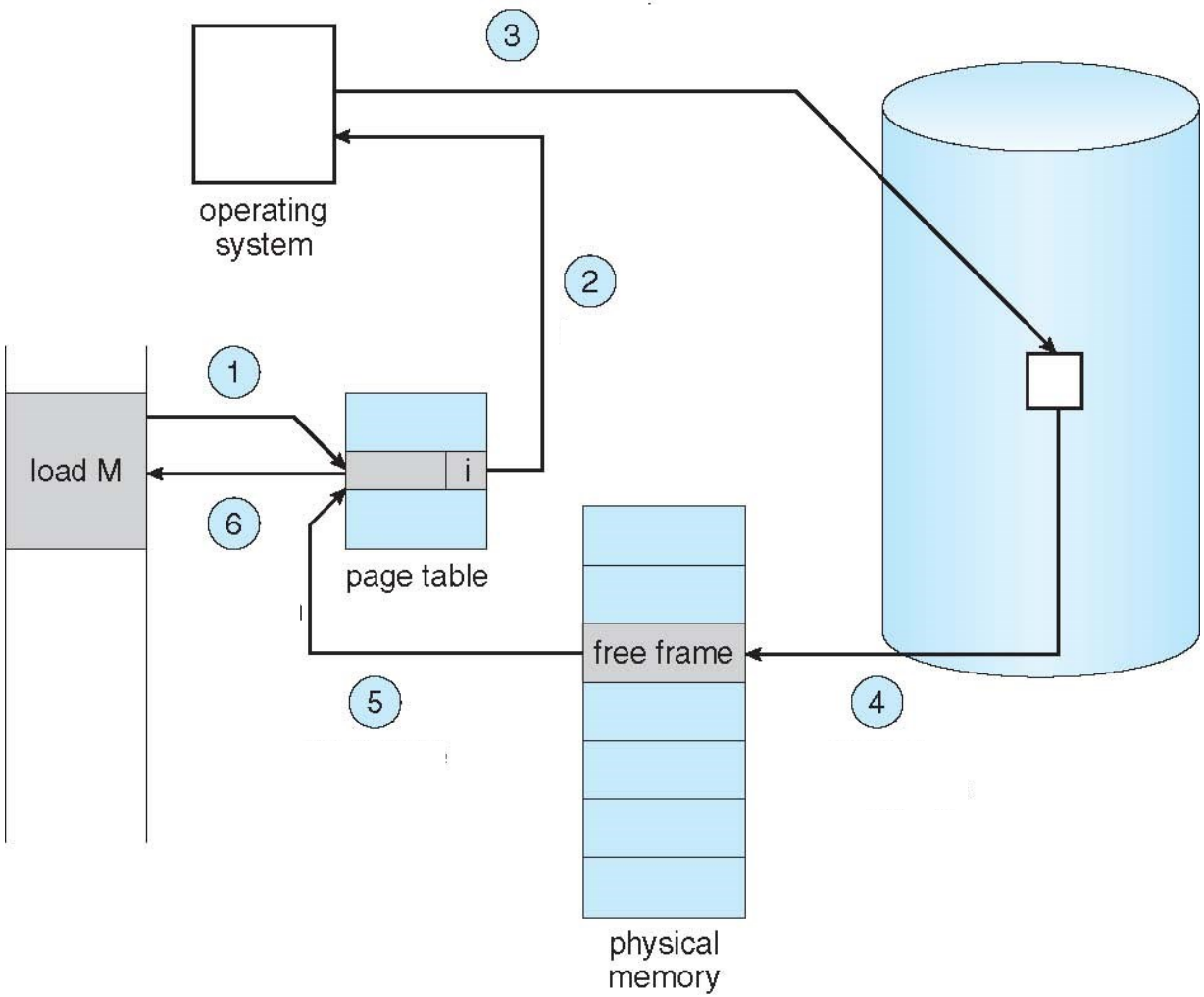
- Зареждане на страница на процес в паметта при необходимост.
- Компонентът, извършващ процеса – ***pager***.



Отказ за страница

- Необходимата страница я няма в паметта -> Page Fault.

Отказ за страница



Отказ за страница (2)

1. ОС преглежда таблицата на страниците
2. Ако е невалидна референция ⇒ abort. Ако е валидна, но не е в паметта - към стъпка (3)
3. Намира свободен фрейм в паметта
4. Зарежда страницата във фрейма чрез планирана I/O операция
5. Модифицира таблиците да указват, че страницата е в паметта (битът за валидиране = v)
6. Рестартира инструкцията, която е довела до отказ за страница

Замяна на страници

Ситуация: в случай на необходимост от зареждане на нова страница, не са налични свободни фреймове в ОП.

Решение: “изхвърляне” на друга страница от ОП и съхраняване на диска.

Базов алгоритъм

1. Намира се позицията на необходимата страница на диска
2. Открива се свободен фрейм:
   - ако е наличен се използва;
   - ако не е се стартира алгоритъм за замяна на страница:
     - избира се определен фрейм за изхвърляне (victim);
     - ако е бил модифициран се записва на диска;
3. Зареждане на необходимата страница във фрейма, обновяване на таблиците на страниците и фреймовете
4. Продължава процеса чрез рестартиране на инструкцията, предизвикала отказ за страница

Стратегии на заместване на страници

Random стратегия:

- Избира се случайна страница за изхвърляне.
- Не е добра стратегия – може да се изхвърли страница, която ще се използва в следващия момент

Стратегии на заместване на страници

FIFO стратегия:

- Всяка страница е маркирана с времето на зареждане в паметта.
- При замяна се избира най-отдавна стоялата в паметта.
- Не добра производителност- страницата може да съдържа модул за инициализация (използван преди много време), но да включва променливи, които се използват много често.

Стратегии на заместване на страници

Last Recently Used (LRU) стратегия:

- Базира се на информация в миналото.
- Заменя се страница, неизползвана за дълъг период от време.
- За всяка страница се поддържа време на последното използване.
- Изисква специален хардуер.

Стратегии на заместване на страници

Least Frequently Used (LFU) стратегия:

- Брой референции към страница.
- Заменя се страница с най-малък брой референции (активно използваните имат голям брой).
- Ситуации при интензивно използвана страница при инициализация на процес, но после рядко използвана.

Стратегии на заместване на страници

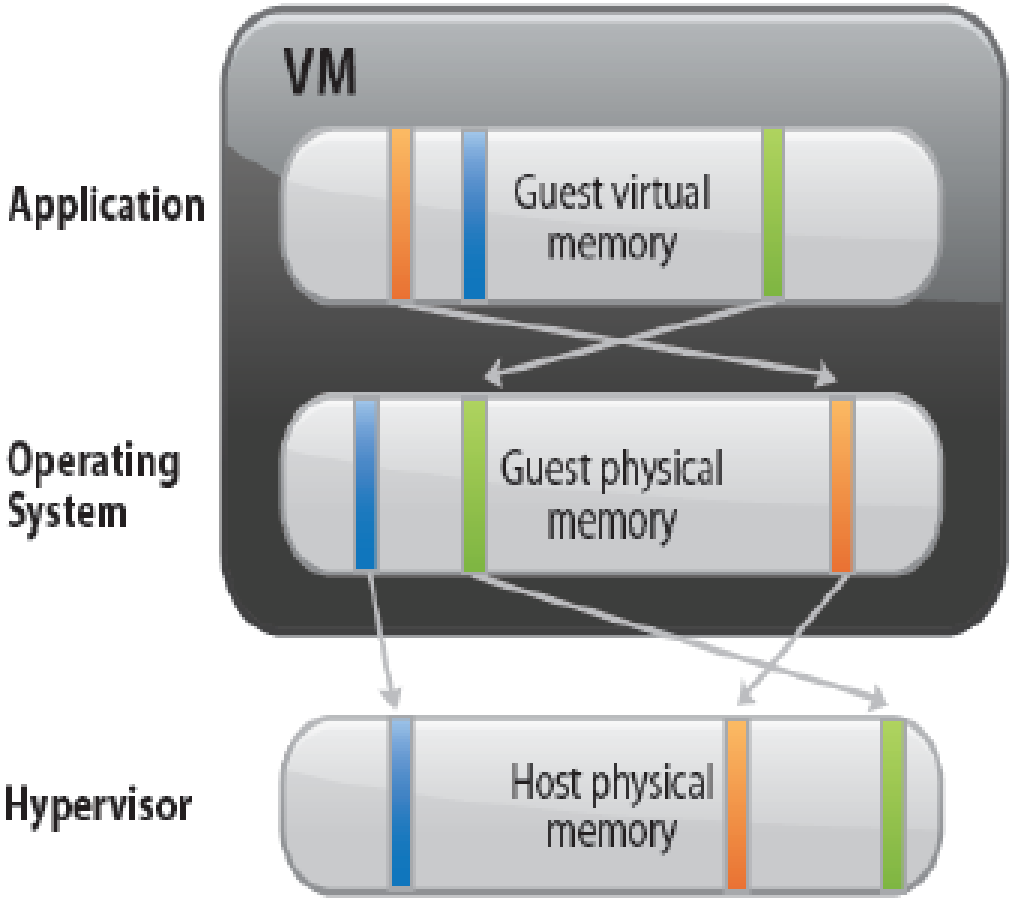
Most Frequently Used (MFU) стратегия:

- Брой референции към страница.
- Заменя се страница с най-малък брой референции (току-що е заредена и още не е използвана).

Виртуализация на паметта

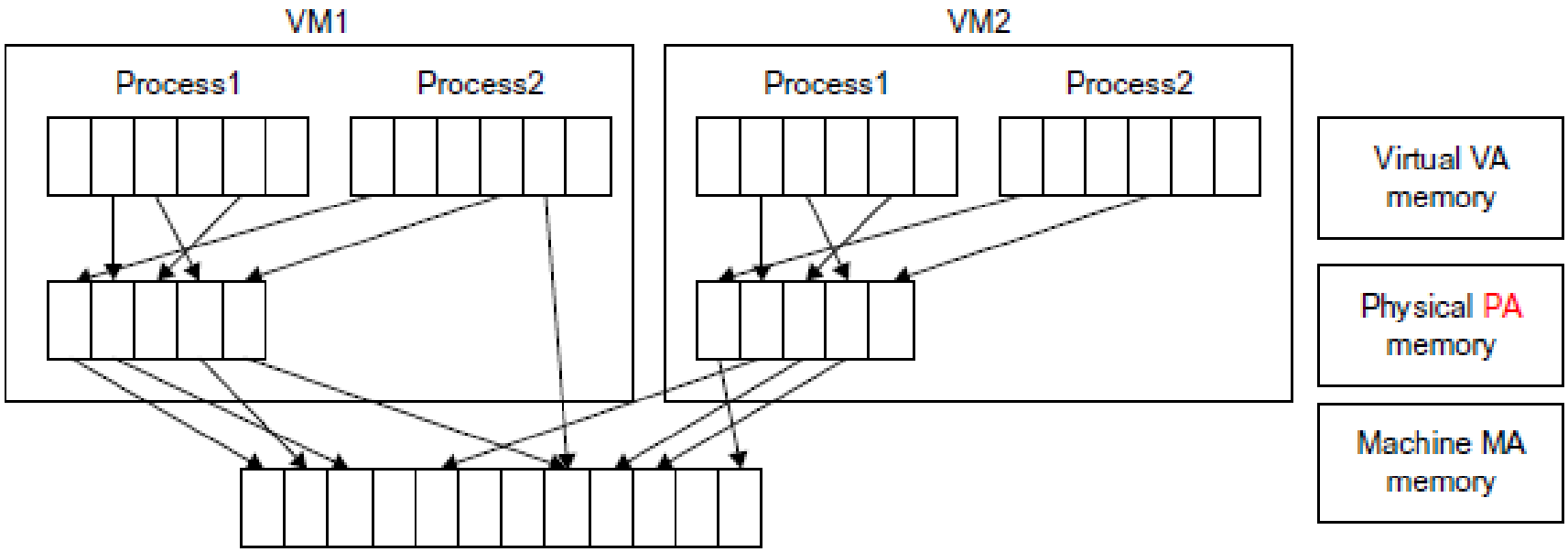
- Паметта на физическата система трябва да бъде споделена и динамично заемана за ВМ;
- По време на изпълнение трябва да се преобразуват логическите адреси на гост-процес към логически адреси на хост-процес.

Виртуализация на паметта



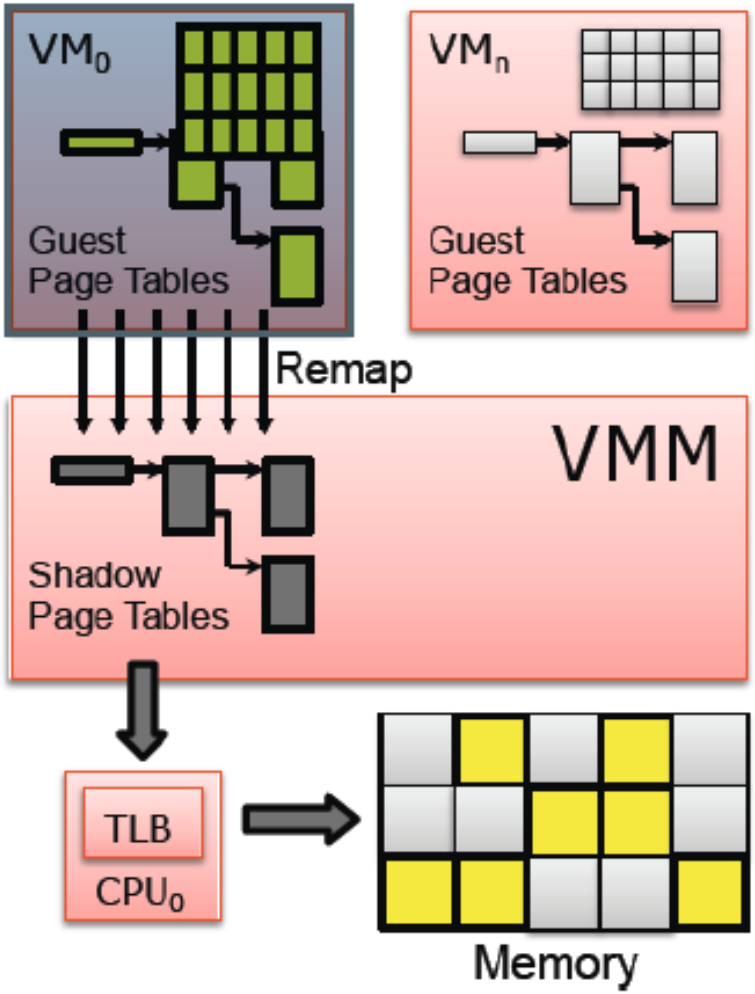
Традиционен подход на преобразуване

- Виртуална памет към физическа памет;
- Физическа памет към машинна памет.



Shadow таблици на страници

- Хипервайзорът прехваща операциите със страниците;
- Изгражда копие на таблиците на страниците.



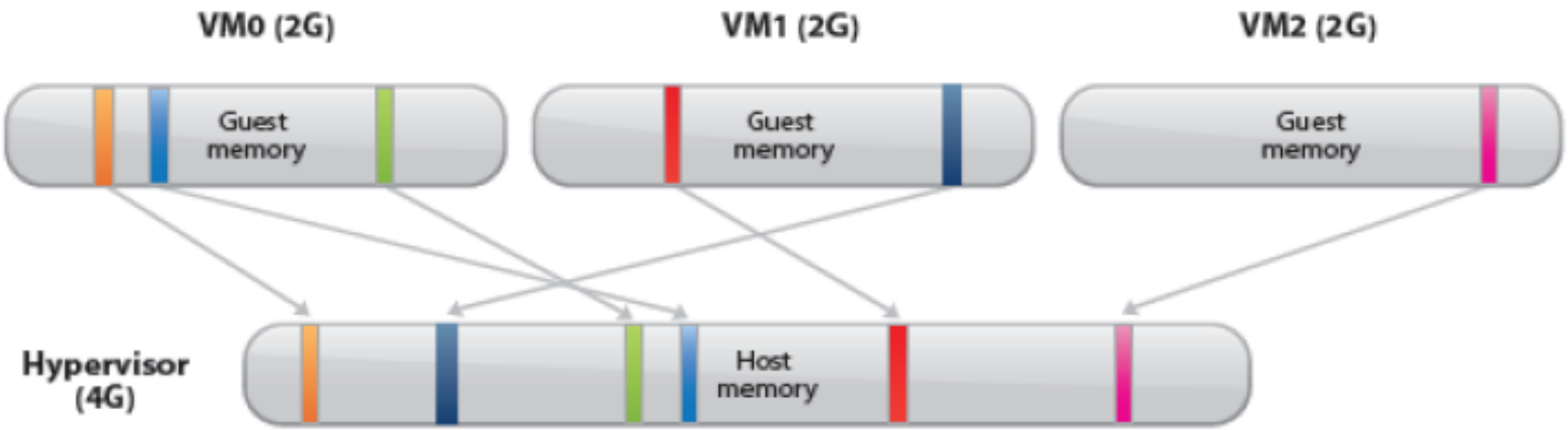
Използвана памет

MemH – използвана хост памет от ВМ
VM_mem_size – обем на паметта на ВМ
VM_mem_over – памет, използвана от хипервайзора за системни данни за ВМ

MemH <= VM_mem_size + VM_mem_over

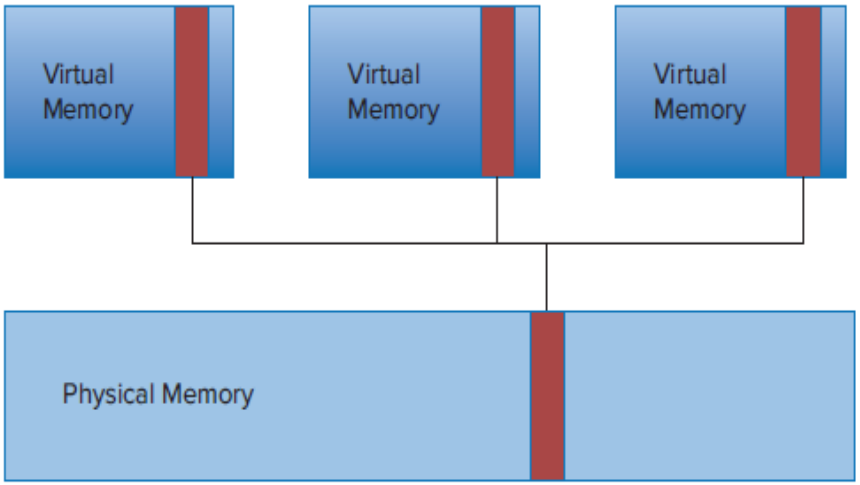
Оптимизация на паметта – memory overcommitment

- Възможността ВМ да използват повече памет, отколкото е наличната физическа памет.



Оптимизация на паметта – споделяне на страници

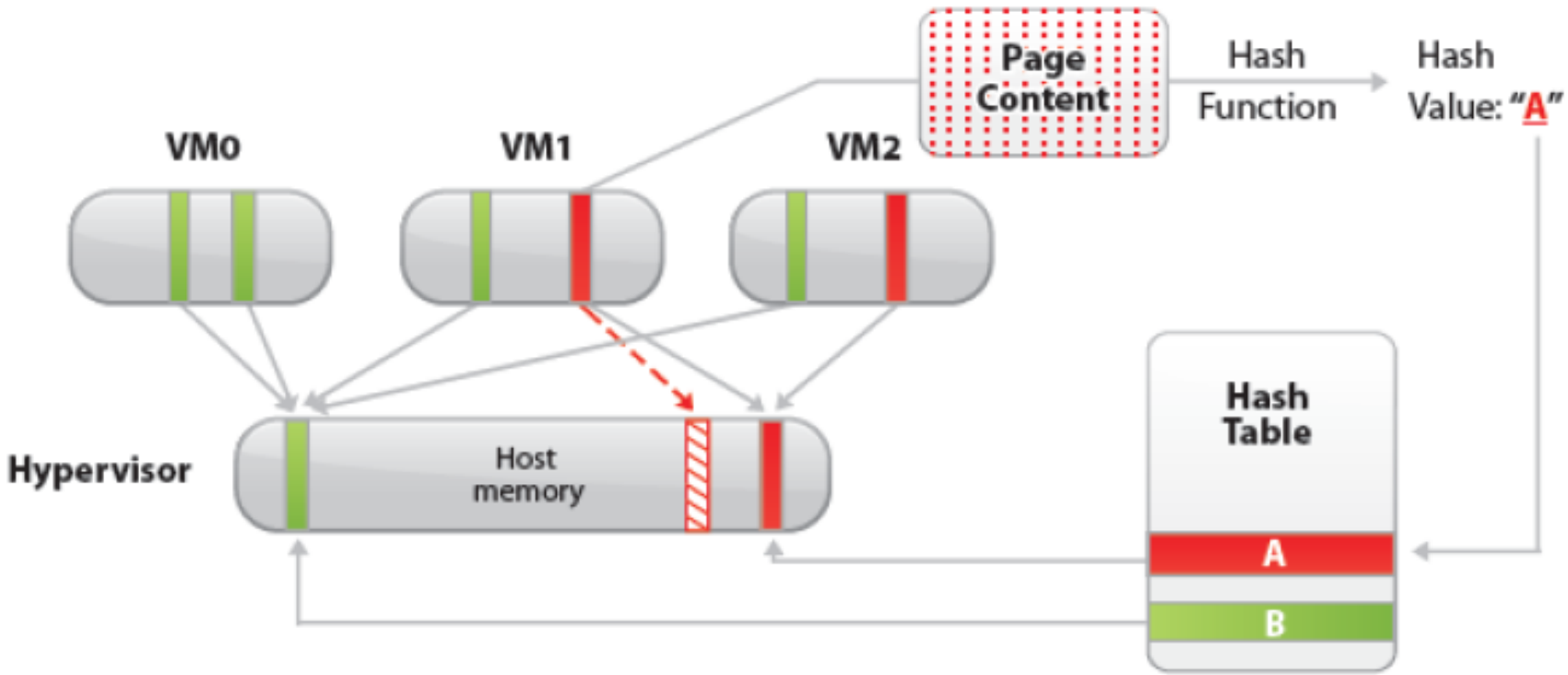
- Някои ВМ могат да имат идентично множество от страници от паметта (едни и същи ОС, приложения или данни);
- Хипервайзорът поддържа в паметта само едно копие на страницата, споделяно между ВМ;
- Ако ВМ модифицира страница, хипервайзорът създава нейно ново копие само за тази ВМ (*Copy-on-Write*).



Context Based Page Sharing Algorithm

- Генерира се хеш на база съдържанието на страница на госта.
- Използва се глобална хеш таблица, съдържаща хеш и номер на физическата страница за съответната споделена;
- При съвпадение на хеша с този в елемент, се извършва пълно сравняване на съдържанието на страницата за избягване на фалшиво съвпадение;
- При потвърждение за съвпадение, съпоставянето между гост-физическа към хост-физическа страница се променя на гост-физическа към споделена хост-физическа страница. Излишното копие в паметта се освобождава.

Context Based Page Sharing Algorithm

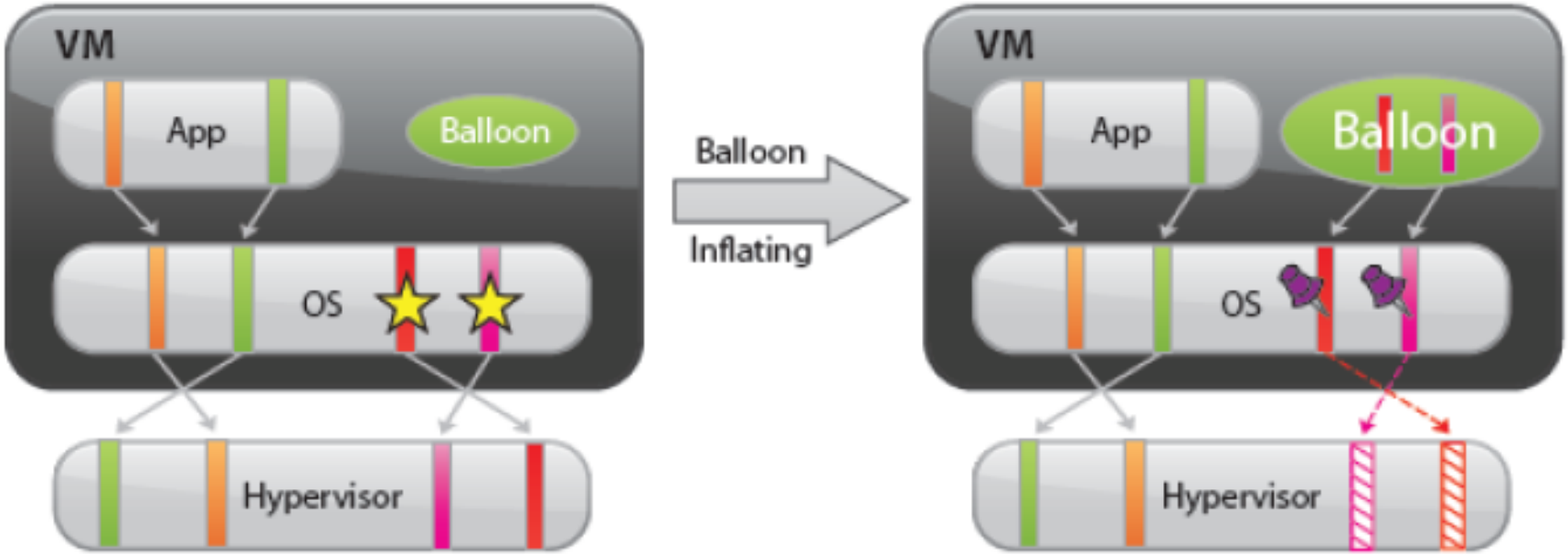


Оптимизация на паметта – ballooning

- При наличие на много ВМ може да настъпи намаляване на свободната памет на хоста;
- ВМ не могат да установят намаляването на паметта на хоста и не могат да освободят физическа памет;

- Използва се специален balloon драйвер, зареден в ОС на госта (*vmmemctl*). Хипервайзорът комуникира с него;
- При необходимост от освобождаване на памет на госта, хипервайзорът изпраща на драйвера съответния размер;
- Драйверът „надува“ чрез заявяване на нови физически страници от госта.

Оптимизация на паметта – ballooning

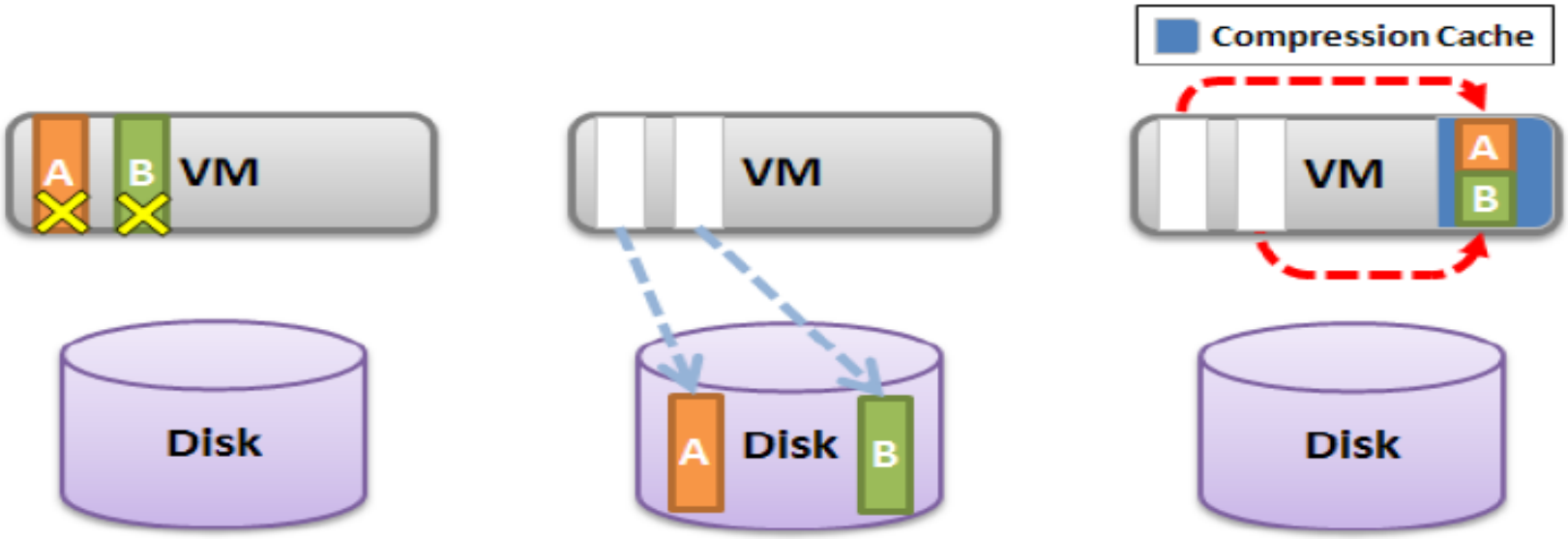


Оптимизация на паметта – Hypervisor swapping

- Ако техниките на ballooning и споделените страници не са достатъчни, хипервайзорът извършва swapping на паметта на госта.
- За всяка ВМ при стартирането й се създава swap файл. Хипервайзорът може директно да изхвърли физическата памет на госта.
- Има влияние на производителността на гост ОС.

Оптимизация на паметта – компресия на паметта

- Идея: изхвърлените страници се компресират и се съхраняват в кеш в паметта.



Управление на паметта при ESX

Предоставят се 3 параметъра за контрол на паметта на хоста:

- **Reservation** – гарантира долната граница на паметта на хоста, резервирана за ВМ, даже и когато паметта му е overcommitted.
- **Limit** – горната граница от паметта на хоста, заета за ВМ.
- **Shares** – право на ВМ да използва част от наличната памет на хоста, базирано на политиката за пропорционално споделено заемане на страници. Например, ВМ с два пъти повече споделени shares спрямо друга има право да използва два пъти повече памет (съобразено с резервацията и лимита).

Управление на паметта при ESX

- При използване заедно на резервиране и лимит, за всяка ВМ да се конфигурира резервираността на 50% от нейния лимит.
- При стартиране на ВМ се създава swap файл с размер лимита минус резервираността.

Управление на паметта при ESX

- Споделените страници (shares) играят важна роля при определяне на заемането на паметта при overcommitting.
- При необходимост от памет, хипервайзорът освобождава памет от ВМ която има малка стойност на отношението *споделени към заети страници*.
- Shares могат да се конфигурират като *low*, *normal*, *high* или произволно:
  - **Low** – 5 shares за 1MB заета памет за ВМ;
  - **Normal** – 10 shares за 1 MB заета памет за ВМ;
  - **High** – 20 shares за 1MB заета памет за ВМ.
- Колкото повече памет се дава на ВМ, толкова повече shares тя получава.

Управление на паметта при ESX

Пример 1

Налични са 5 ВМ, всяка с 2GB заета памет и *share=normal*. Хостът има само 4GB физическа памет. Всяка ВМ получава 20000 shares (10 * 2000).

Общо shares са 5 * 20000 = 100000.

Всяка ВМ ще получи еднаква share 20000/100000=1/5 от наличните ресурси – 4000/5=800MB.

Управление на паметта при ESX

Пример 2

Нека едната ВМ е с *share=high*. Тя получава 40000 shares (20 * 2000).

Общата сума на shares става 120000.

Тази ВМ ще получи 40000/120000=1/3 от наличните ресурси, т.е. 4000/3-1333MB.

Останалите ВМ ще получат само 20000/120000=1/6 от наличните ресурси, т.е. 4000/6=666MR всяка.

Управление на паметта при ESX

- Ограничение на чистия алгоритъм за пропорционално споделяне е, че не се взема под внимание актуалното използване на паметта от ВМ. Някои ВМ с висока стойност на shares може да имат неизползвана памет, докато други ВМ с по-малка да имат недостиг.
- Решението е да се „наказват“ ВМ с повече неизползвана памет за сметка на тези с активно използвана.
- Ако ВМ не използва своята памет, нейното съотношение *споделени към заети страници* се намалява.
- Ще се освобождава памет преференциално за ВМ, които не използват напълно своята памет.

Въпроси?