Оперативни системи и рачунарске мреже

Александар Картељ
aleksandar.kartelj@gmail.com
Рачунарска гимназија

Меморија

- Један од основних ресурса рачунарског система
- Било који физички уређај који привремено или трајно чува податке
  - Регистри
  - Кеш меморија
  - Примарна меморија
  - Секундарна меморија

Регистри

- Меморија уграђена у процесор
- Већ смо споменули неке регистре:
  - Акумулатор (AC)
  - Инструкциони регистар (IR)
  - …
- Ово је најбржа меморија, али је има јако мало, пар KB
- А истовремено и најближа процесору (унутар њега)
- Стога је укупно време приступа јако кратко

Кеш меморија

- Више меморије него што чине регистри, неколико MB
- Међутим, нешто спорија је од регистарске меморије
- Служи да надомести велику разлику у брзинама између регистарске меморије и примарне меморије

Примарна (унутрашња, главна) меморија

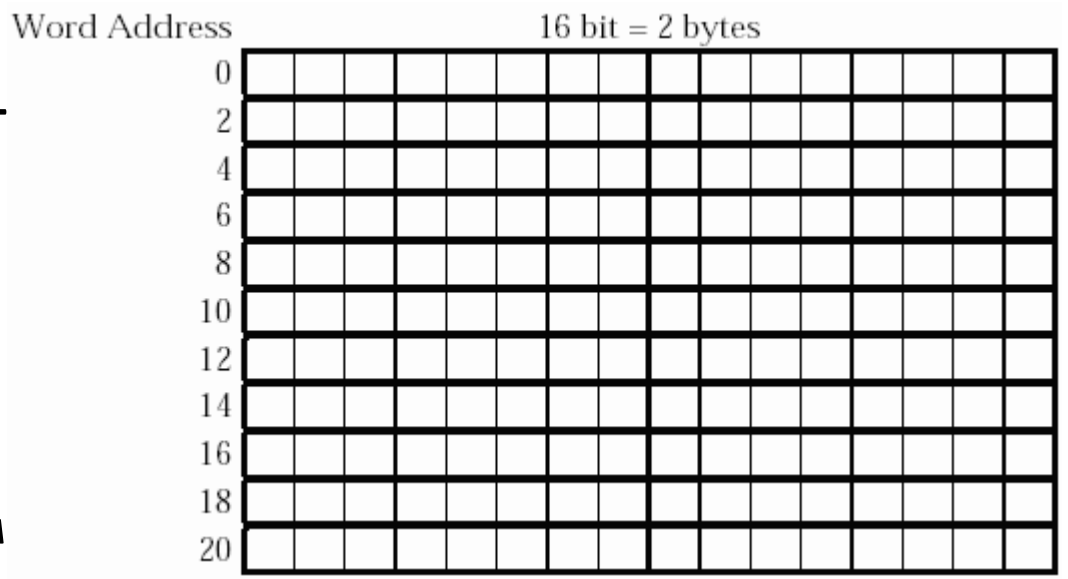
- Меморија која садржи инструкције и податке које оперативни систем тренутно користи
- Спорија је од кеш меморије, али је има више, нпр. 8GB
- Сачињена је махом из два типа меморија:
  - RAM меморија – меморија са случајним приступом
    - Ова меморија није доступна када је рачунар угашен, односно губи садржај приликом гашења
  - ROM меморија – меморија само за читање
    - Ова меморија је доступна и када је рачунар угашен, обично се користи за подешавање BIOS-а

Секундарна (спољашња) меморија

- Не губе садржај приликом гашења рачунара
- Намена им је да запамте податке
- Приликом рада рачунара, често се програми или подаци учитавају из секундарне меморије у примарну меморију
- Примери овакве меморије су:
  - Хард диск
  - CD-ROM
  - DVD
  - …
- Ова меморија је најспорија, али је има највише

Структура меморије

- Меморије се обично представљају мат
  - Редови представљају меморијске речи
    - Обично је величина речи 8, 16, 32, 64 бита, односно 1, 2, 4, 8 бајтова
  - Свака меморијска реч има своју адресу
    - Адреса представља редни број у матрици
    - Број меморијских речи, зависи од величине м
      нпр. ако имамо 1GB:
      - То може бити 125 милиона речи величине 8 бајтова
      - Или милијарду речи величине 1 бајт
    - Колико нам треба битова да би записали адресу (адресирали) речи на позицији милијарда?
    - Колико меморије можемо да адресирамо са 32 бита?



Учитавање програма у меморију

- Програми који се извршавају и њихови подаци се морају налазити у главној меморији
  - У том смислу се морају прекопирати из секундарне меморије, нпр. Хард диска у главну меморију
- Где у главну меморију копирамо програм при покретању?
  - Да ли је локација увек иста?
- У модерним ОС, та локација варира, јер се у датом тренутку више програма (процеса) може налазити у меморији
- Не желимо ни за један од њих да фиксирамо простор у меморији заувек, зашто?

Технике управљања меморијом

- Једна од тема управљања меморијом јесте проблем учитавања, односно на који начин се програми учитавају, избацују итд.
- Постоје неколико техника управљања:
  1. Фиксно партиционисање
  2. Динамичко партиционисање
  3. Партнерски систем
  4. Страничење
  5. Сегментација – само описно
  6. Виртуелна меморија – (прескочићемо због сложености)

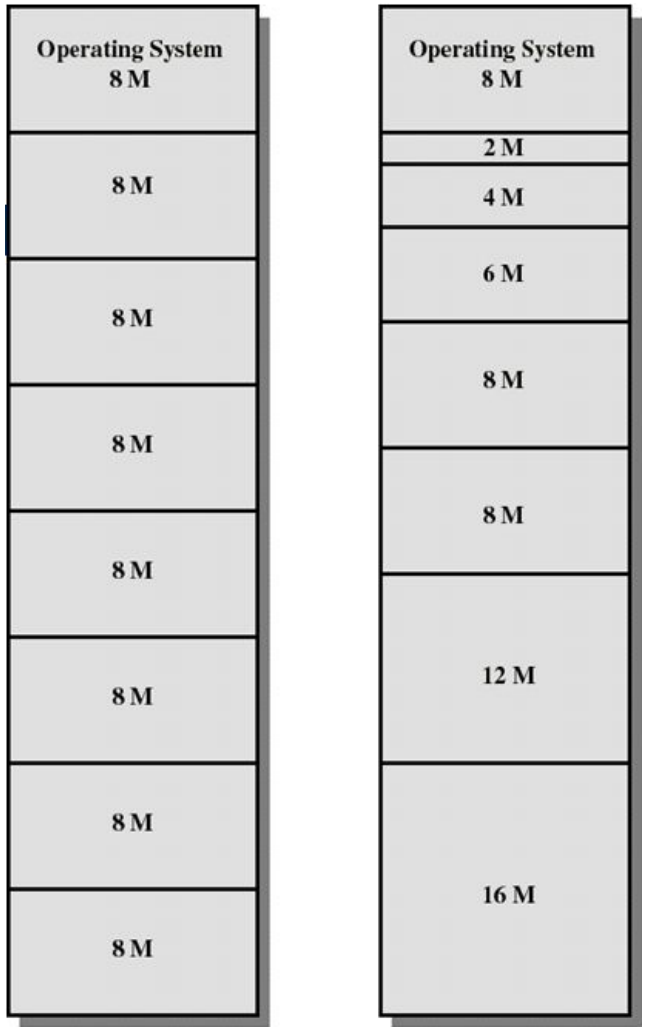
Фиксно партиционисање

- Меморија се дели на партиције (делове) фиксне величине
- Сваки процес чија величина мања или једнака величини партиције може да се учита у партицију
- Ако су све партиције пуне, а треба да се изврши неки процес, ОС избацује процес из неке пуне и убацује други процес
- Постоје две врсте фиксног партиционисања:
  - На партиције једнаке величине
  - На паратиције различите величине (обично расту са фактором 2)

Фиксно партиционисање (2)

- Фиксно партиционисање је неефикасно
- Јавља се *унутрашња фрагментација*
- Унутрашња фрагментација је простор који остаје неискоришћен у оквиру партиције, када је програм мањи од партиције
- У коју партицију бисте сместили програм:

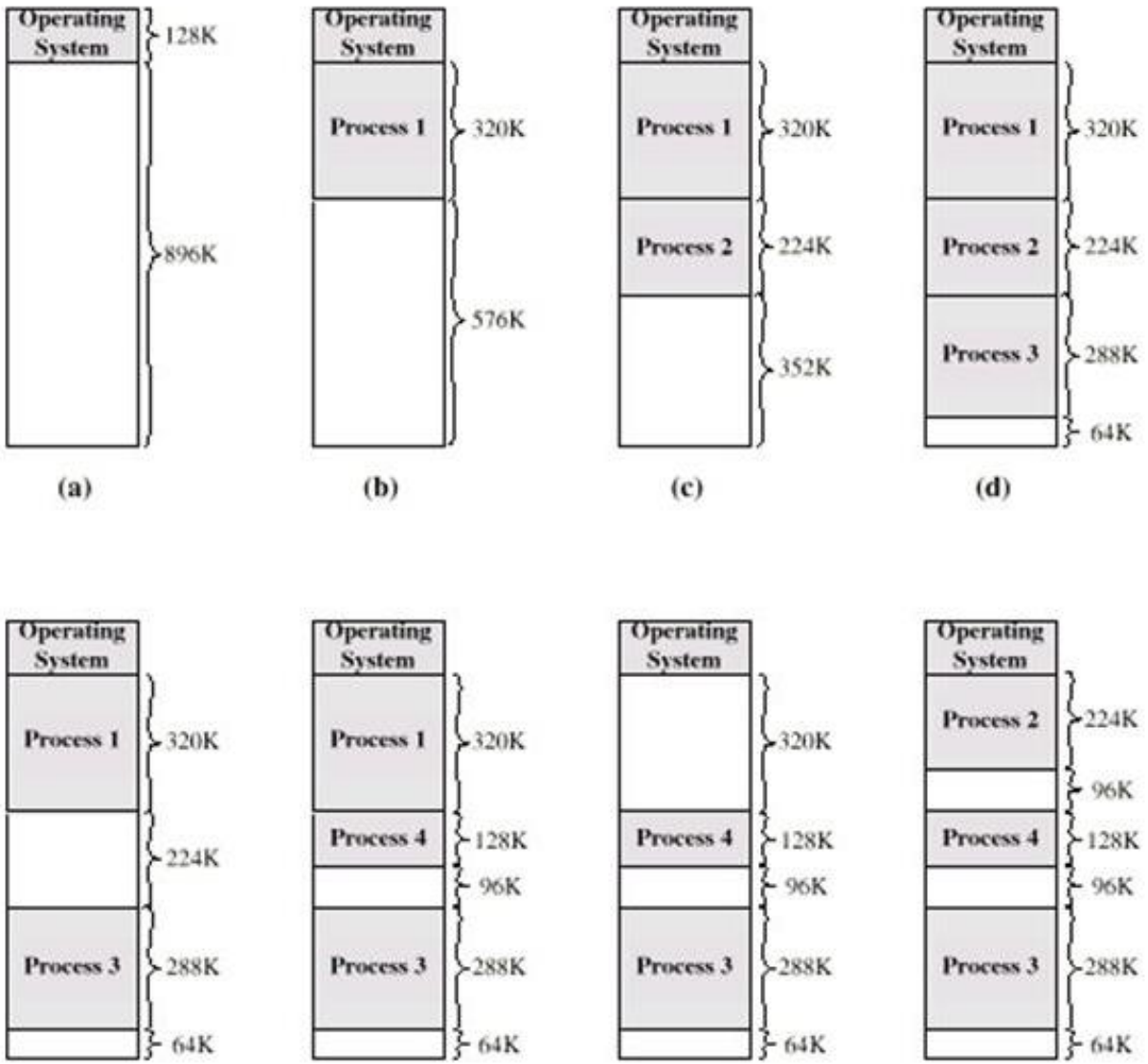
1. Када користите партиције исте величине?
2. Када су партиције различите величине?



Динамичко партиционисање

- Партиције су променљиве величине
- Када се процес учита у меморију,
  додељује му се тачно колико му треба
- Међутим, како пролази време, процеси се избацују из меморије
  и на тај начин остављају *рупе* – зашто је ово проблем?
- Овај процес се назива *спољашња фрагментација меморије*
- Спољашња фрагментације се превазилази сажимањем
  - Повремено померање делова меморије како би се од више мањих рупа направила једна довољно велика да у њу стане нови процес

Ефекат динамичког партиционисања



Динамичко партиционисање (2)

- Алгоритам убацивања програма:
  1. Најбољи одговарајући
     - Бира партицију која му је најближа по димензијама
  2. Први одговарајући
     - Бира прву која је довољно велика
  3. Следећи одговарајући
     - Као први одговарајући, само почиње тражење од места последњег убацивања
- Који је најгори од ових, шта мислите?

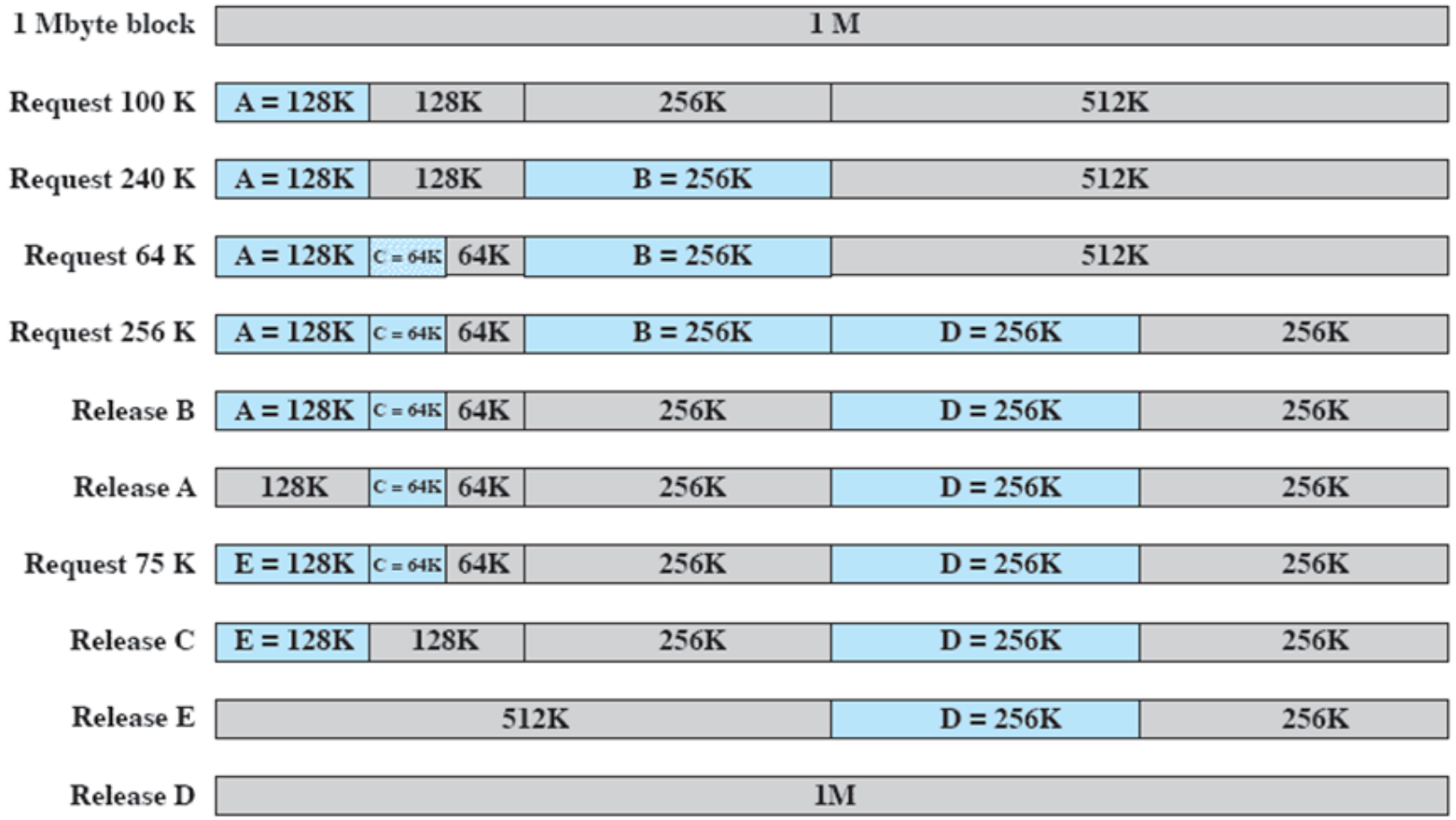
Динамичко партиционисање (2)

- Алгоритам убацивања програма:
  1. Најбољи одговарајући
     - Бира партицију која му је најближа по димензијама
  2. Први одговарајући
     - Бира прву која је довољно велика
  3. Следећи одговарајући
     - Као први одговарајући, само почиње тражење од места последњег убацивања
- Који је најгори од ових, шта мислите?
  - Најгори је „најбољи одговарајући“, зато што изазива спољашњу фрагментацију са најмањим рупама, које се после не могу искористи ни за шта
  - Друга два алгоритма су слична, али је први одговарајући најефикаснији...

Партнерски систем

- Блокови меморије су степени двојке, нпр. од 64K до 1MB
- На почетку, све је један велики простор од 1MB
- Ако јави захтев од нпр. 100K, простор се преполовљава, све док се не дође до величина која је тек мало већа од 100K, тј. до 128K
- За сваки следећи захтев се ради слично:
  - Ако постоји слободан регион који је „најбољи одговарајући“, искористи га
  - У супротном дели неки већи регион на два дела све док не добије „најбољи одговарајући“
- Демонстрација је на следећем слајду

Партнерски систем (2)

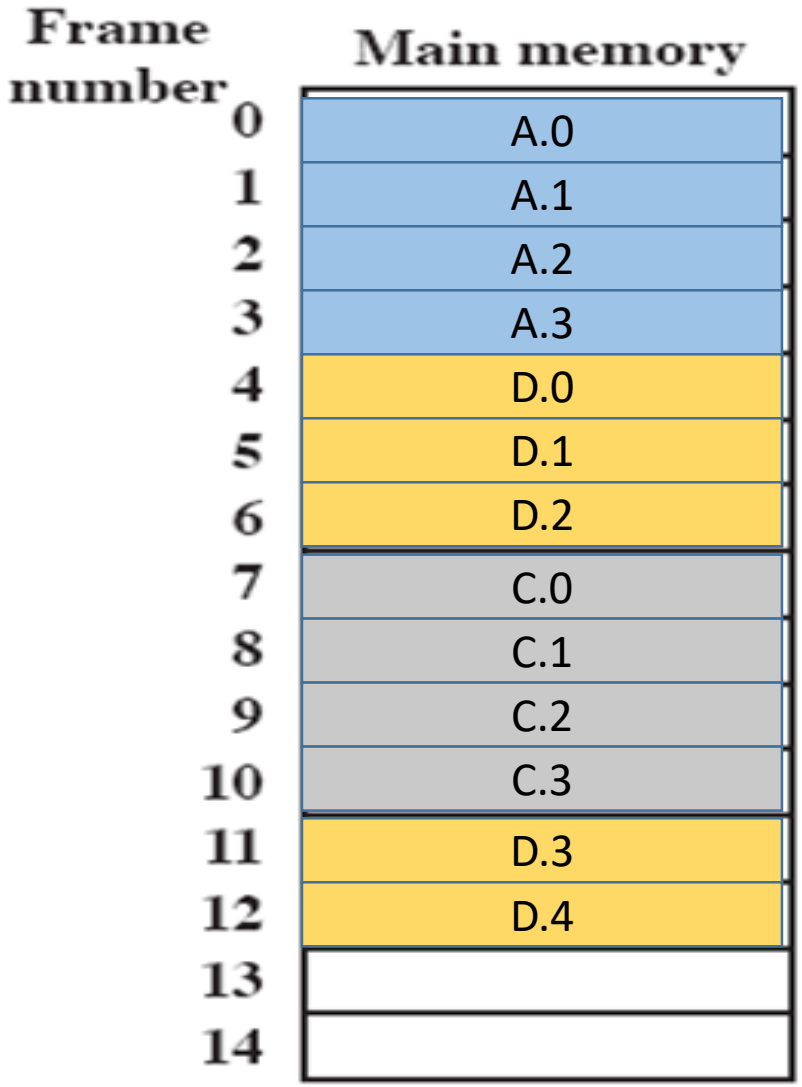


Страничење

- Главна меморија издељена на једнаке делове фиксне величине
  - Ти делови су релативно мали, нпр. 4KB и називају се оквири
- Меморија потребна процесу се такође издели на такве делове
  - Делови су исте величине као и оквири и називају се странице
- Надаље, када се процес учитава:
  - За сваку његову страницу се проналази одговарајући оквир
  - Информације о вези оквира и страница чува тзв. Табела страница

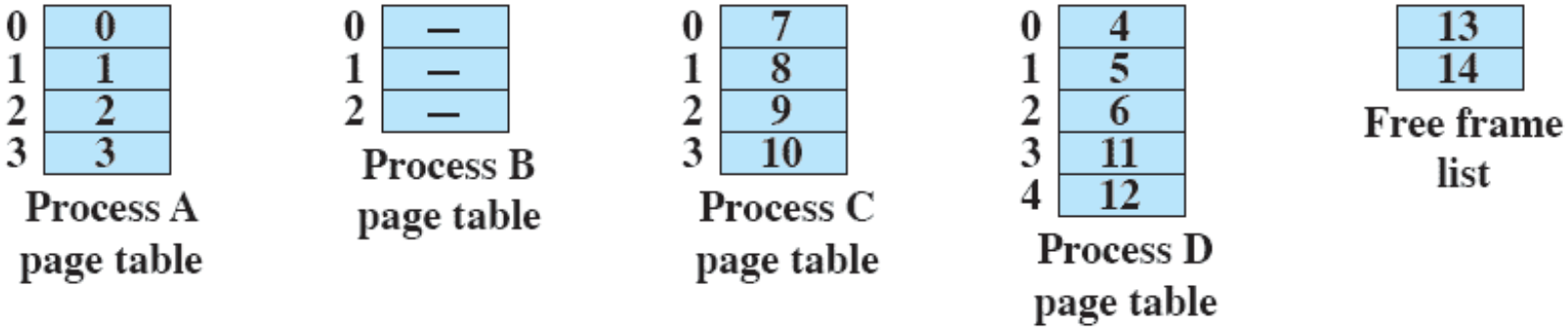
Пример страничења

- Сценарио:
  - Процес А тражи 4 странице и добија их
  - Процес В тражи 3 и добија их (оквири 4,5,6)
  - Процес С тражи 4 странице, добија их
  - У међувремену процес В завршава
    - Притом оставља „рупу“ на оквирима 4, 5 и 6
  - У међувремену стиже захтев процеса D
    - Он тражи 5 оквира и добија их,
      само они сада нису узастопни
    - Додељују се оквири 4, 5, 6, 11 и 12



Табела страница

- Сваки процес памти табелу страница, односно списак додељених оквира
- Постоји посебна табела слободних оквира на нивоу ОС-а



Страничење - карактеристике

- Страничење елиминише проблем спољне фрагментације
  - Сваки ослобођени оквир се може искористити,
    јер сви процеси користе страницу исте величине
- Такође значајно умањује унутрашњу фрагментацију
  - Може постојати део неискоришћеног заузетог простора
    само у последњој страници
- Шта су предности, а шта мане повећавања или смањивања величина страница односно оквира?

Сегментација

- Сваком процесу се додељују различити сегменти меморије
- Сваки такав сегмент припада одвојеном делу меморијског простора
- Стандардни сегменти су:
  - Сегмент за код програма – инструкције програма
  - Стек сегмент – подаци које користи програм
  - Хип сегмент – подаци које користи програм
  - Сегмент за глобалне податке – дељен између више процеса
  - Сегмент за библиотеке – такође дељен између више процеса
  - Итд.
- Распоређивање слично као код партиционисања,
  само је овде ОС свестан логичке употребе меморије