GERÊNCIA DE MEMÓRIA

INF009 – Laboratório de Sistemas Operacionais

Agenda

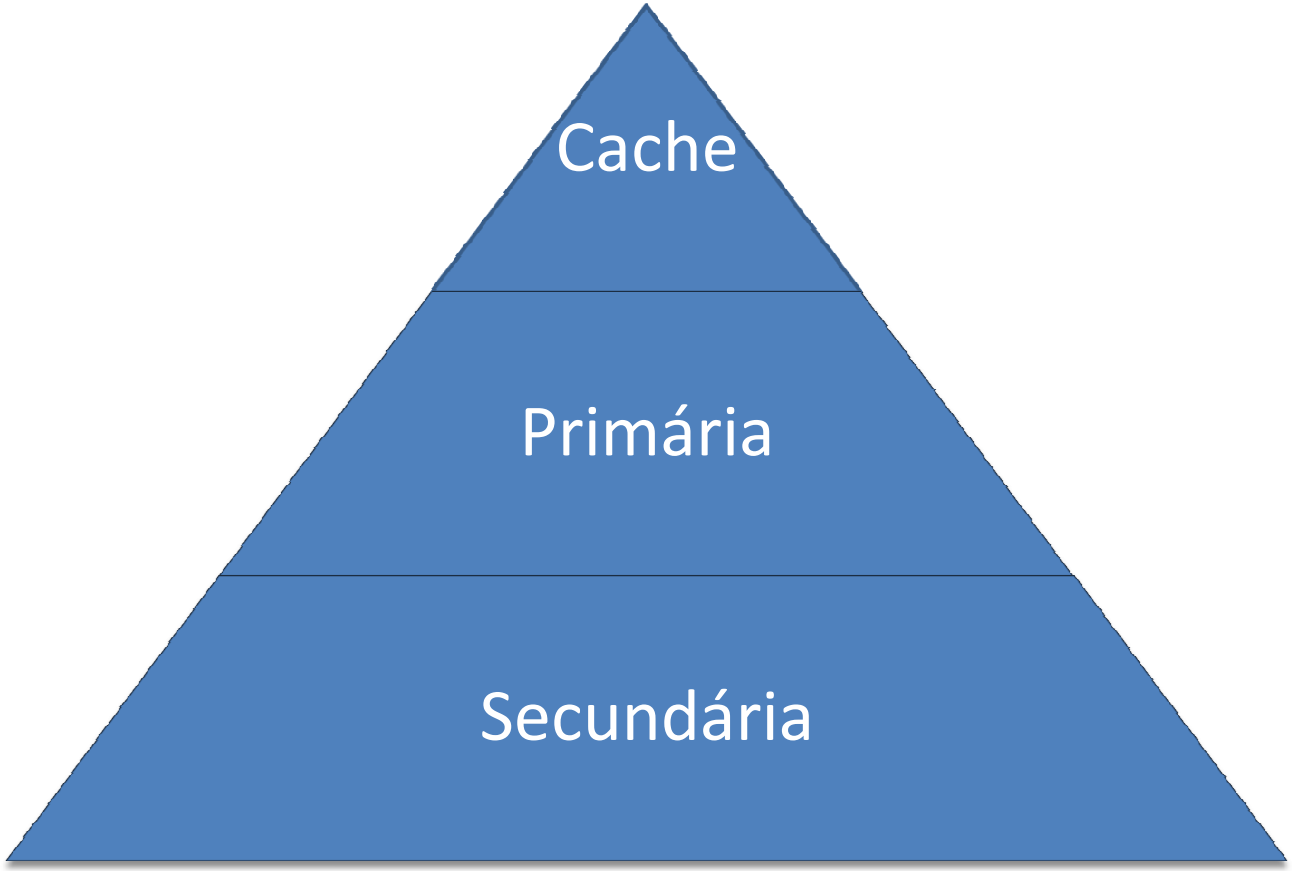
- Motivação
- Gerência de memória com monoprogramação
- Gerência de memória com multiprogramação
  - Partições Fixas
    - Divisão e Alocação das Partições
    - Proteção
  - Troca de Processos (Swapping)
  - Endereçamento de memória
  - Memória Virtual com paginação
    - Algoritmos de Substituição de Páginas
  - Memória Virtual com Segmentação

Motivação

- Memória é um recurso escasso
  - Programas crescem mais rápido
  - “Os programas tendem a crescer ocupando toda a memória principal” Parkinson
  - Anos 80, VAX (4MB) era compartilhado por dezenas de usuários
- Os programas só executam se estiverem na memória principal
- Infelizmente não existe a memória perfeita (rápida, infinita e não volátil)
- Otimizar o uso da memória principal

Hierarquia de Memórias

- Relembrando...



Imprtância da Multiprogramação

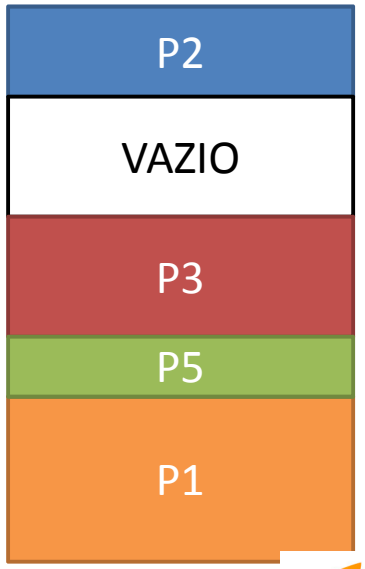
- Multiprogramação otimiza o uso do processador
- Se você tiver um único programa, a cada acesso a disco (por exemplo) o processador fica parado
- Note que a utilização do processador vai depender diretamente do número de processos e do perfil dos mesmos (cpu-bound ou io-bound)

Monoprogramação

- Sem troca de processos ou paginação
  - Só existe o sistema operacional e um programa
  - O usuário recebe o prompt, executa o programa e volta para a tela do prompt
  - U tilizado apenas em sistemas embarcados simples
  -

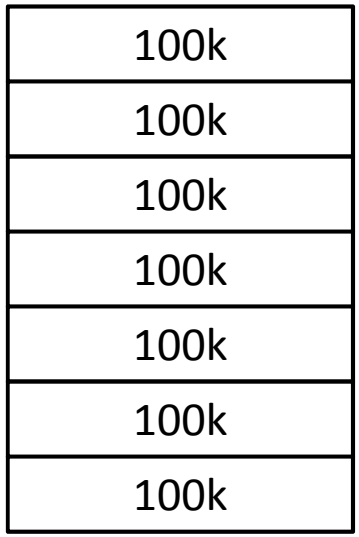
Multiprogramação

- Diversos processos compartilham o processador
- Memória contém dados de mais de um processo
- Realidade dos sistemas atuais



Multiprogramação com partições fixas

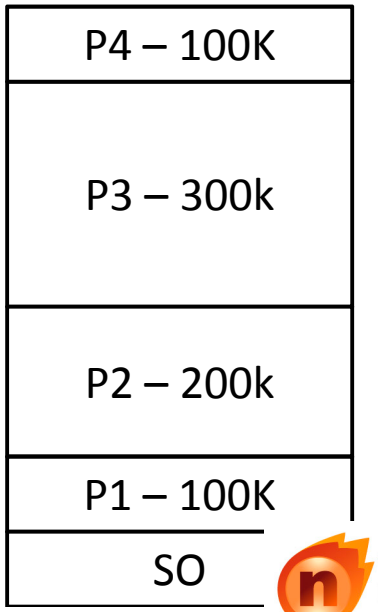
- Maneira mais simples de utilizar a memória
- Particionar em n partições
  - De tamanhos iguais?
  - Qual o tamanho?



  - Observe que neste modelo o prograador deve conhecer o hardware e detalhes do SO

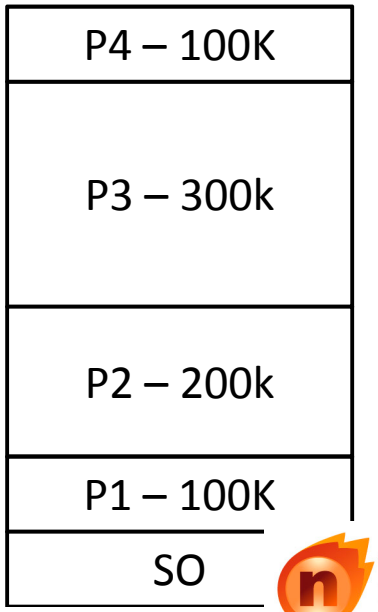
Multiprogramação com partições fixas

- Mais inteligente utilizar partições de diferentes tamanhos
- Mas que partição escolher na chegada de cada processoou programa?



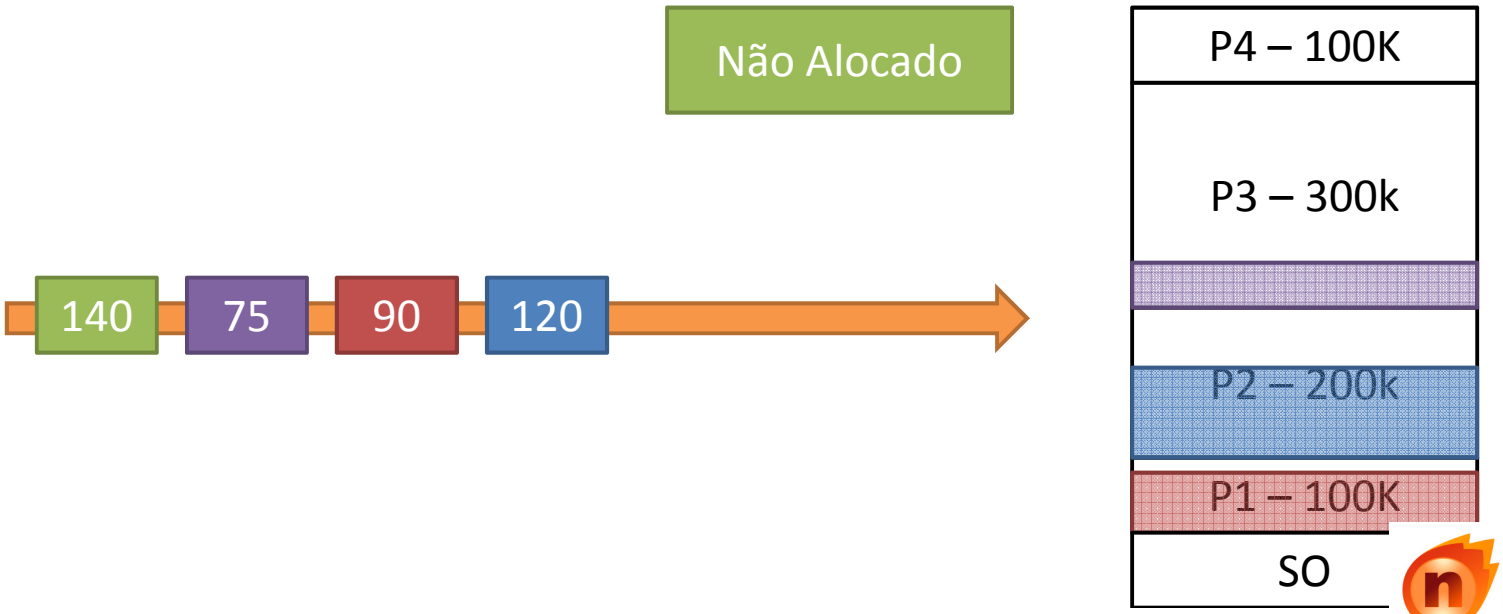
Multiprogramação com partições fixas

- Algoritmos
  - First Fit
  - Next Fit
  - Best Fit
  - Worst Fit



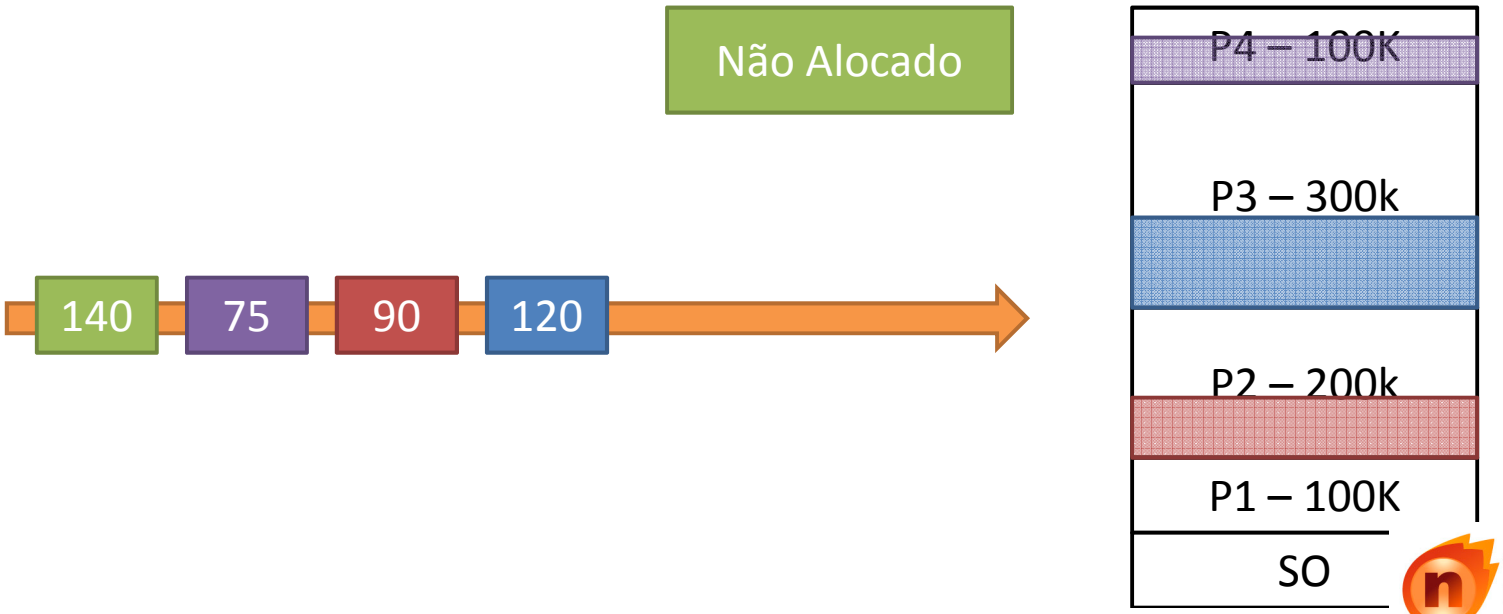
Multiprogramação com partições fixas

- First Fit
  - Aloca na primeira posição que o processo cabe
  - Algoritmo simples
  - Baixa eficiência



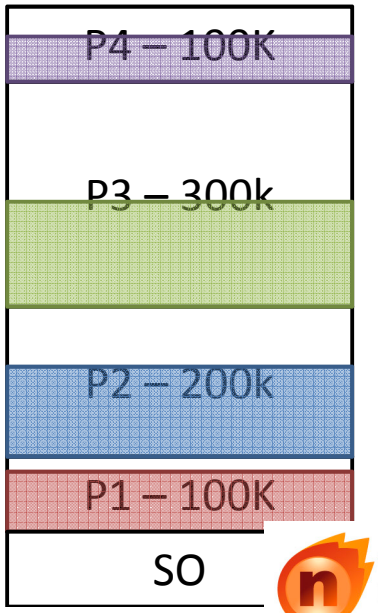
Multiprogramação com partições fixas

- Next Fit
  - Aloca na segunda posição que o processo cabe
  - Mais rápido (por que?)
  - Também não é eficiente



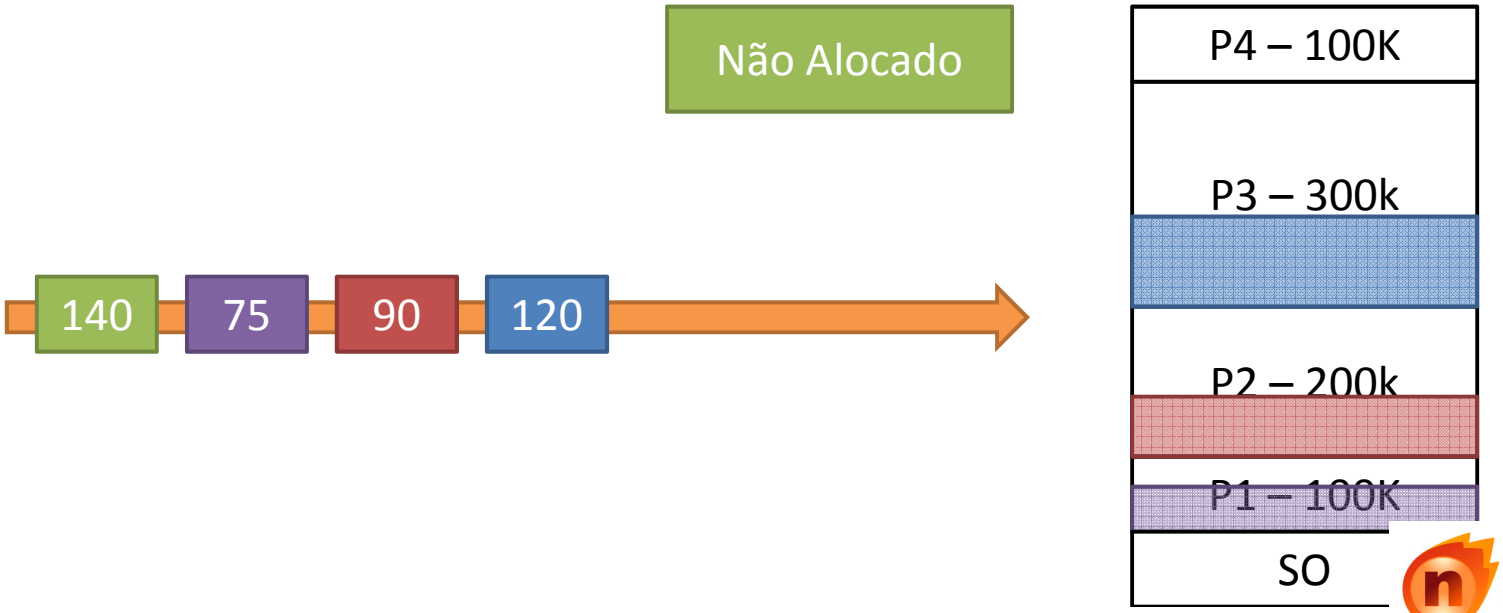
Multiprogramação com partições fixas

- Best Fit
  - Busca o que retornará o menor espaço vazio
  - Mais Eficiente (espaço utilizado)
  - Mais Complexo
  - O que acontece se os processos solicitarem mais memória?



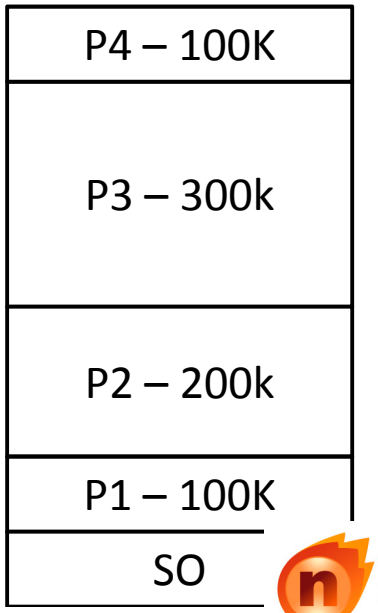
Multiprogramação com partições fixas

- Worst Fit
  - Busca o que retornará o maior espaço vazio
  - Bom quando a pilha de execução e dados crescem muito
  - Na prática tem baixa eficiência



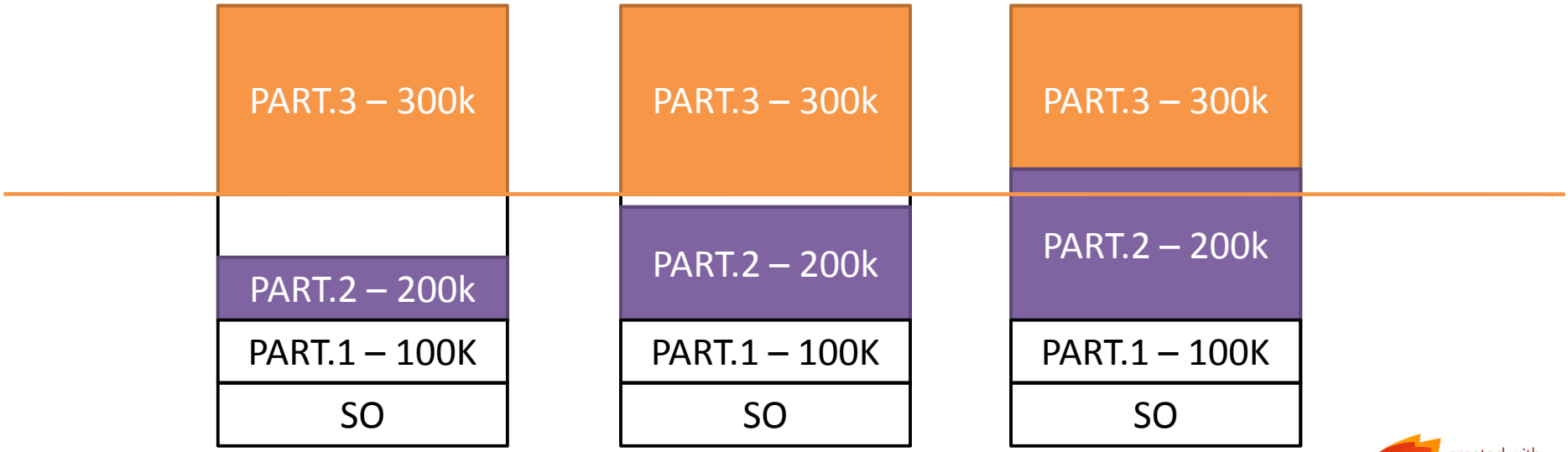
Multiprogramação com partições fixas

- Outra forma de pensar...
- Utilizar múltiplas filas com best fit
  - Neste caso como fica a partição P3?



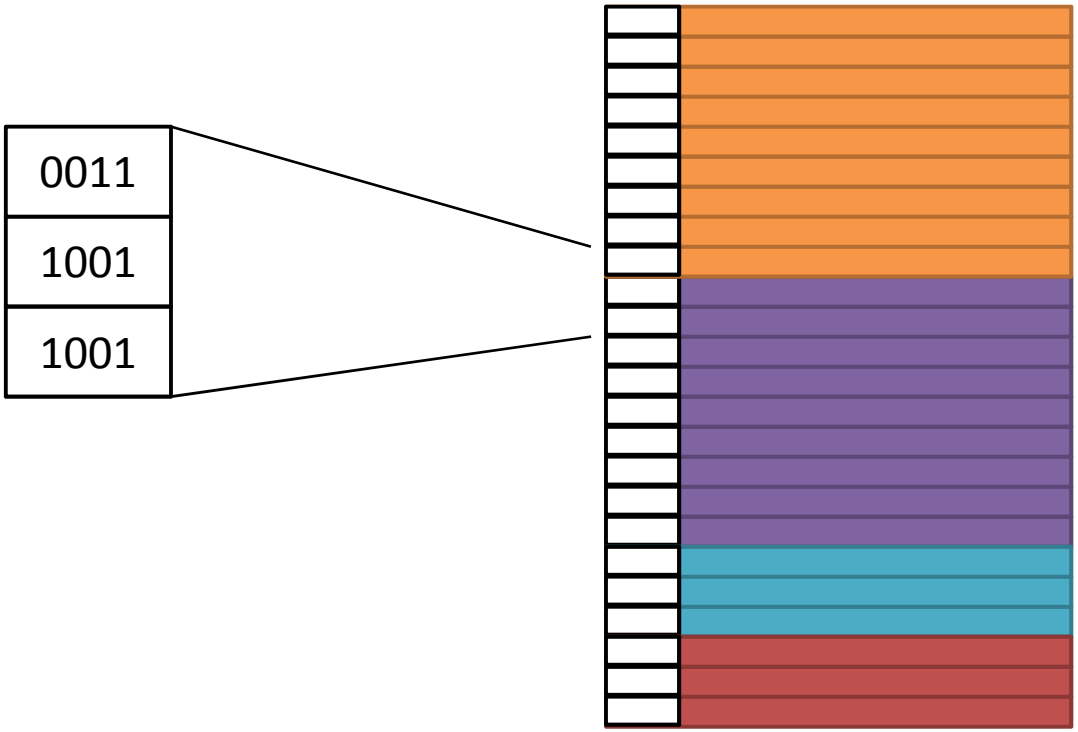
Multiprogramação com partições fixas

- Problema
  - Se o processo alocado na partição 2 começar a crescer de forma a invadir a partição 3?



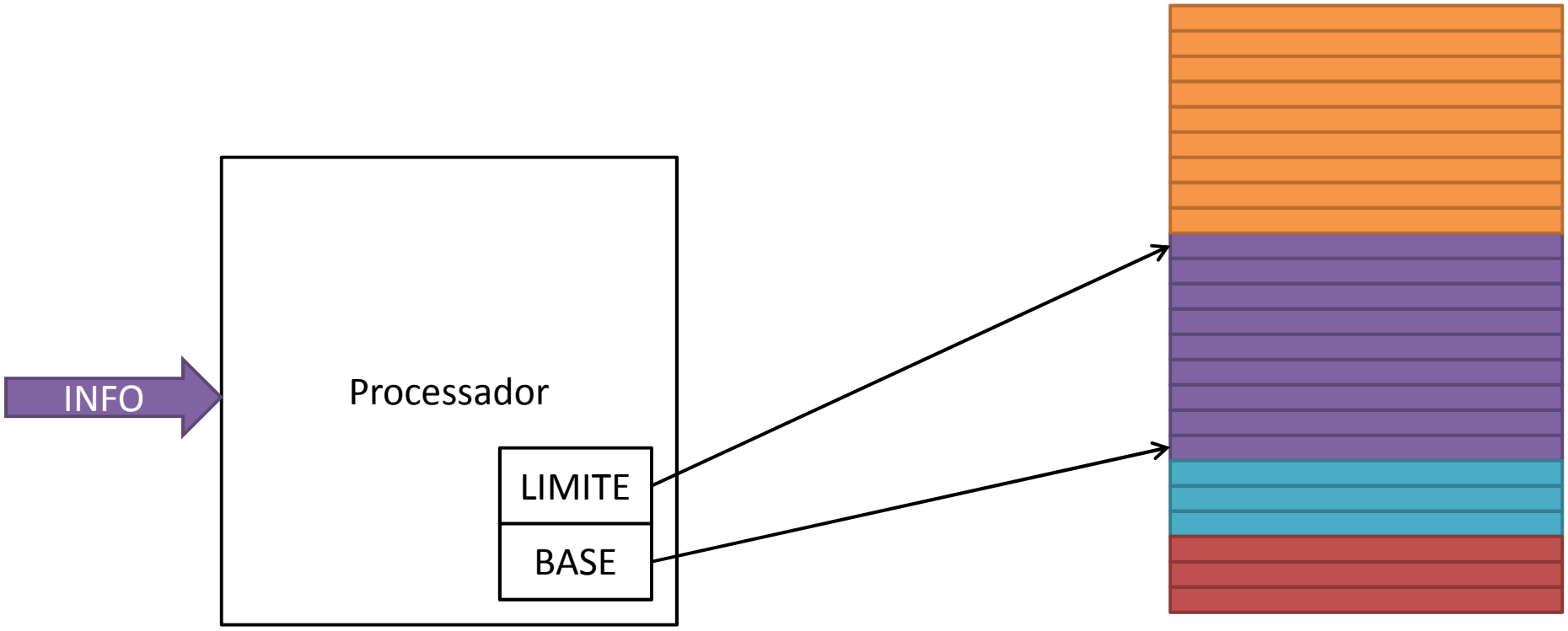
Multiprogramação com partições fixas

- Solução 1 - Proteção
  - IBM (IBM 360) Quebrou a memória em blocos de 2kB e separou 4 bits para identificar o processo



Multiprogramação com partições fixas

- Solução 2 – Registrador Base e Limite
  - Quando um processo é escalonado, também é informado a partição do mesmo



Multiprogramação sem partições fixas

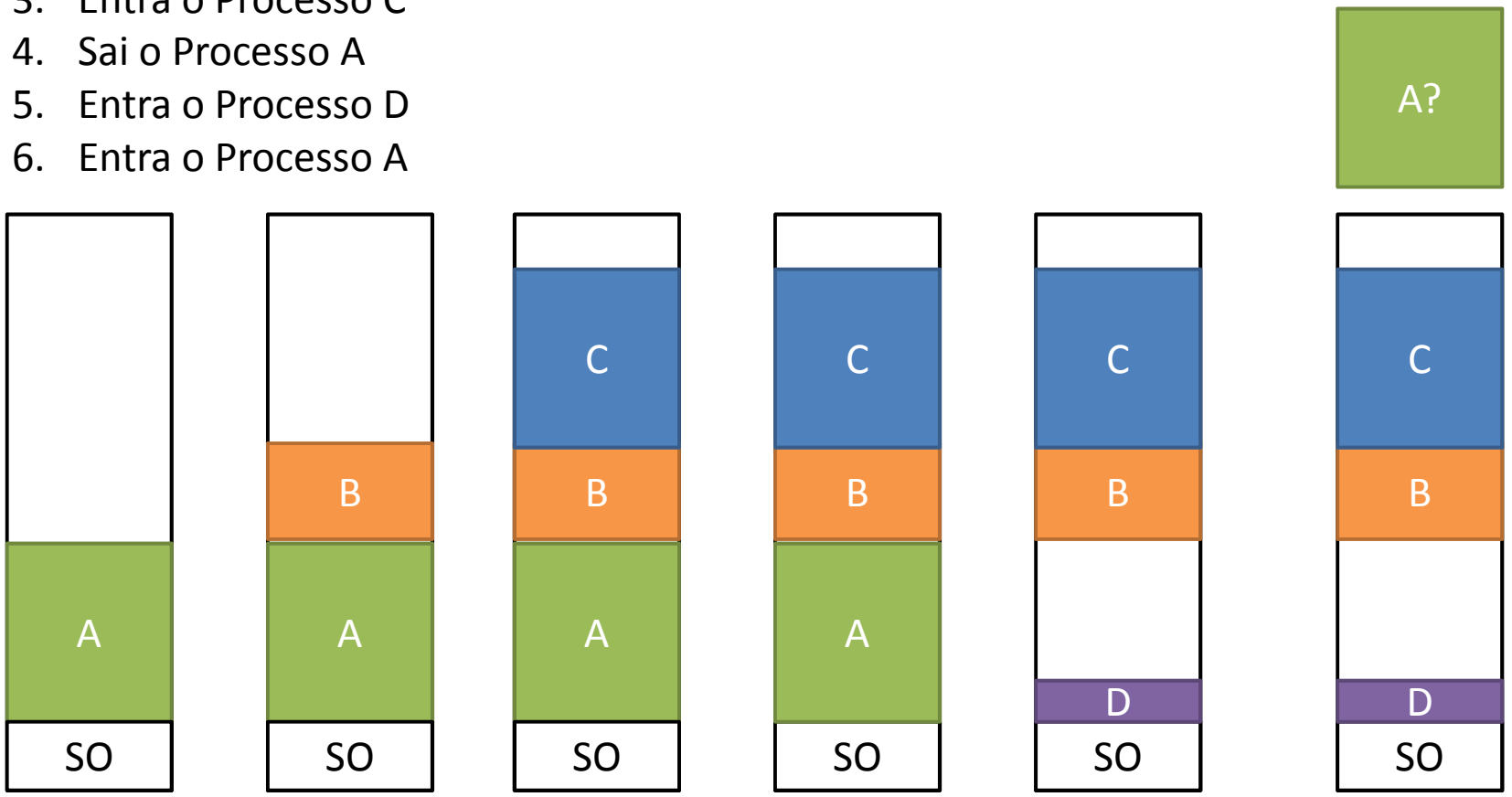
- Como funcionaria a troca de processos?
- Como saber o início e fim de cada processo na memória?

Troca de Processos

- Não há memória principal para todos os programas
- Swapping (troca de processos)
  - Consiste em trazer um programa todo para a memória, executá-lo e devolvê-lo para o disco
  - Apenas com Swapping, os programas são carregados totalmente na memória principal

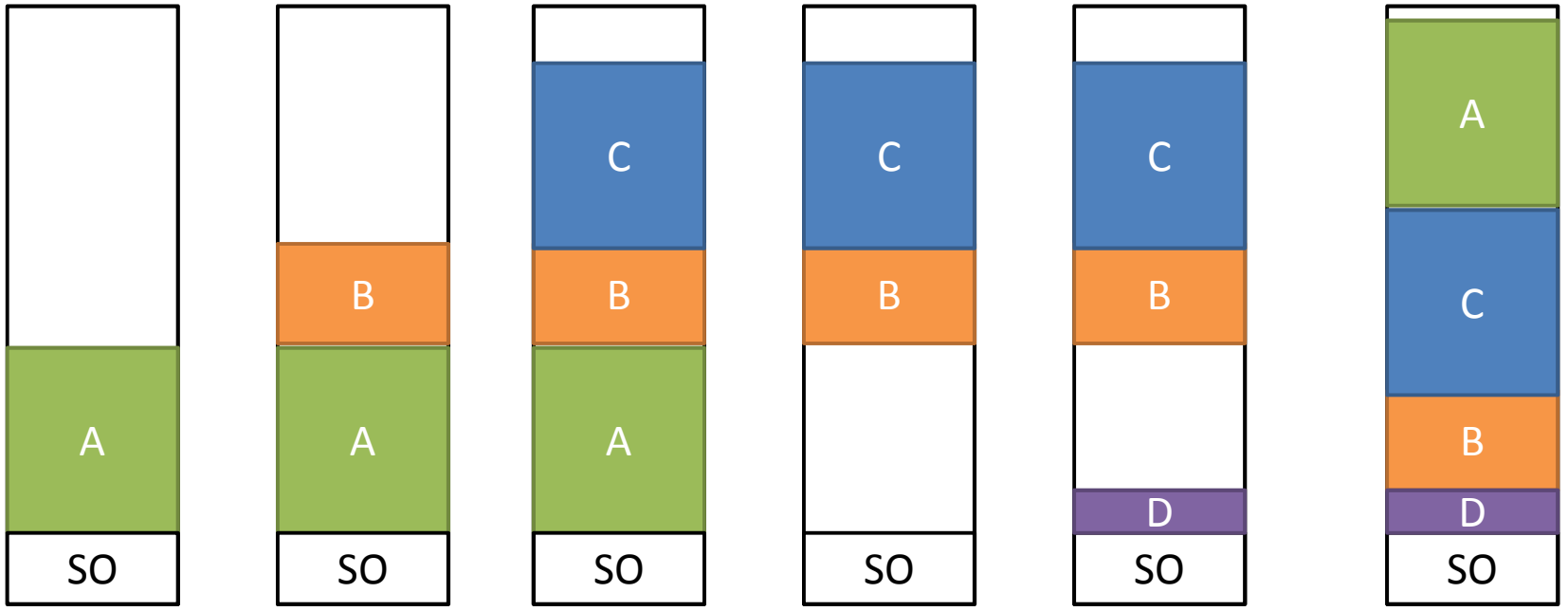
Swapping

1. Entra o Processo A
2. Entra o Processo B
3. Entra o Processo C
4. Sai o Processo A
5. Entra o Processo D
6. Entra o Processo A



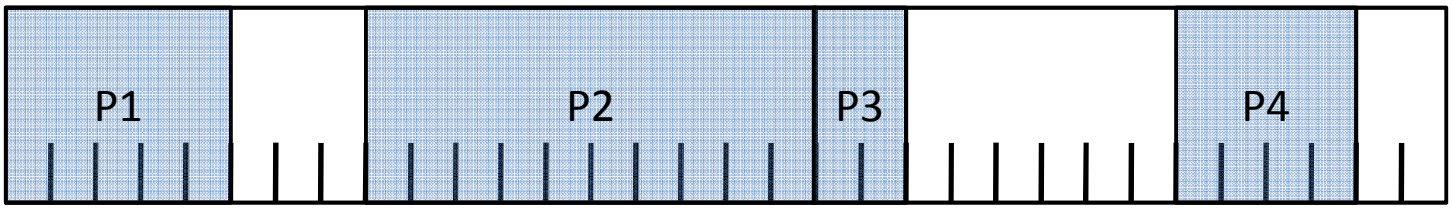
Swapping

- Compactação de Memória



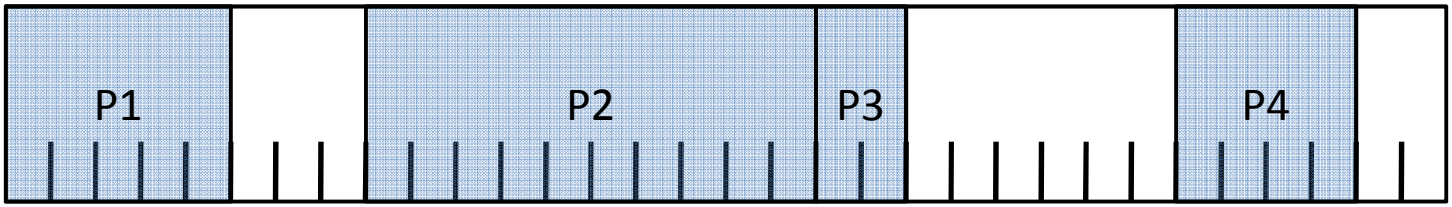
Endereçamento de Memória

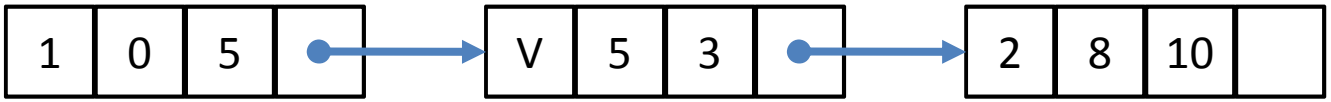
- O SO deve gerenciar as posições da memória na alocação dinâmica
  - Agora as partições não são fixas
  - Informações de cada processo
  - Espaços Vazios
  - Que estrutura de dados utilizar?
  - Exemplo esta memória tem 32 blocos de 2K



Endereçamento de Memória

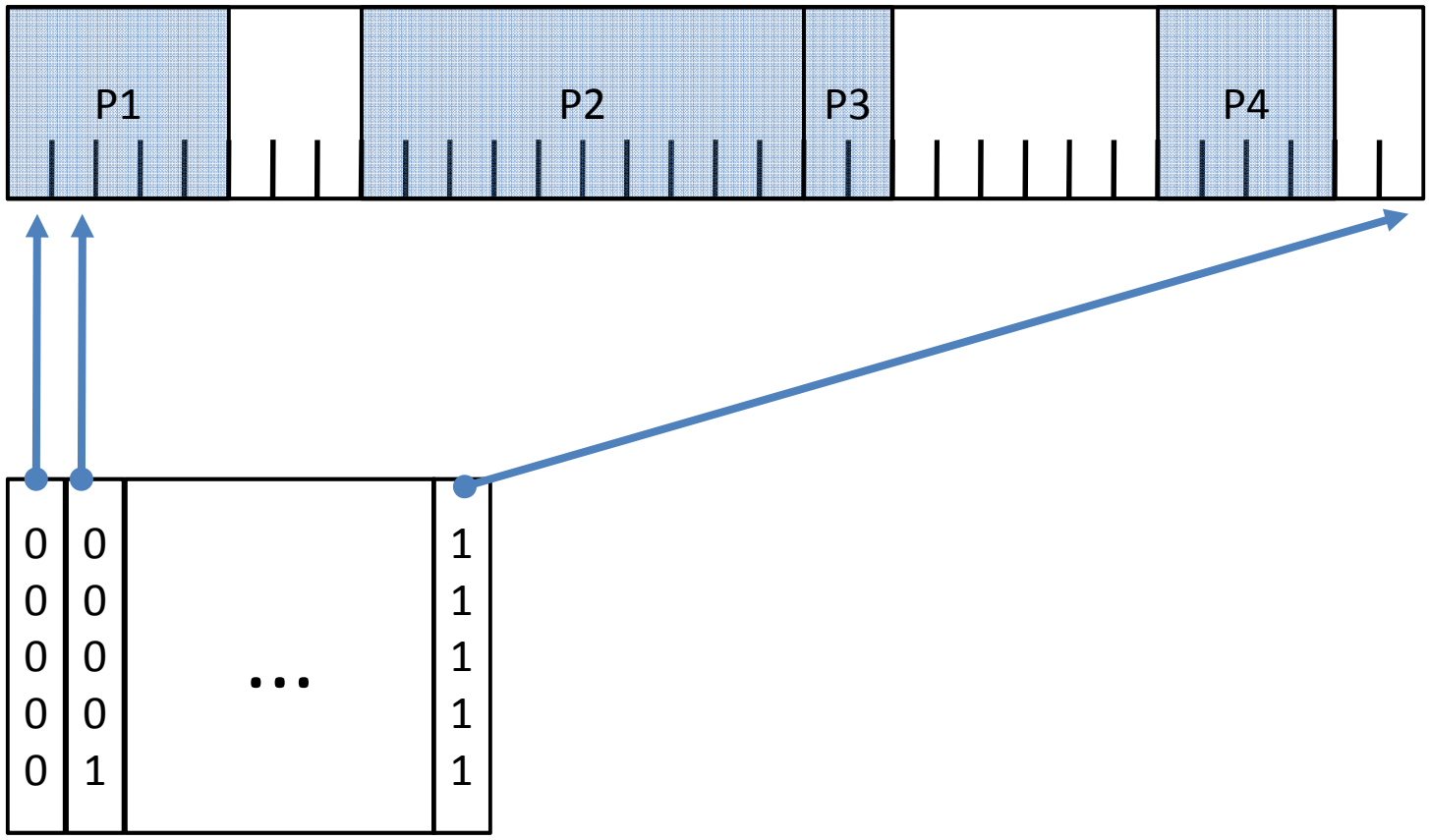
- Lista encadeada





Endereçamento de Memória

- Mapa de bits

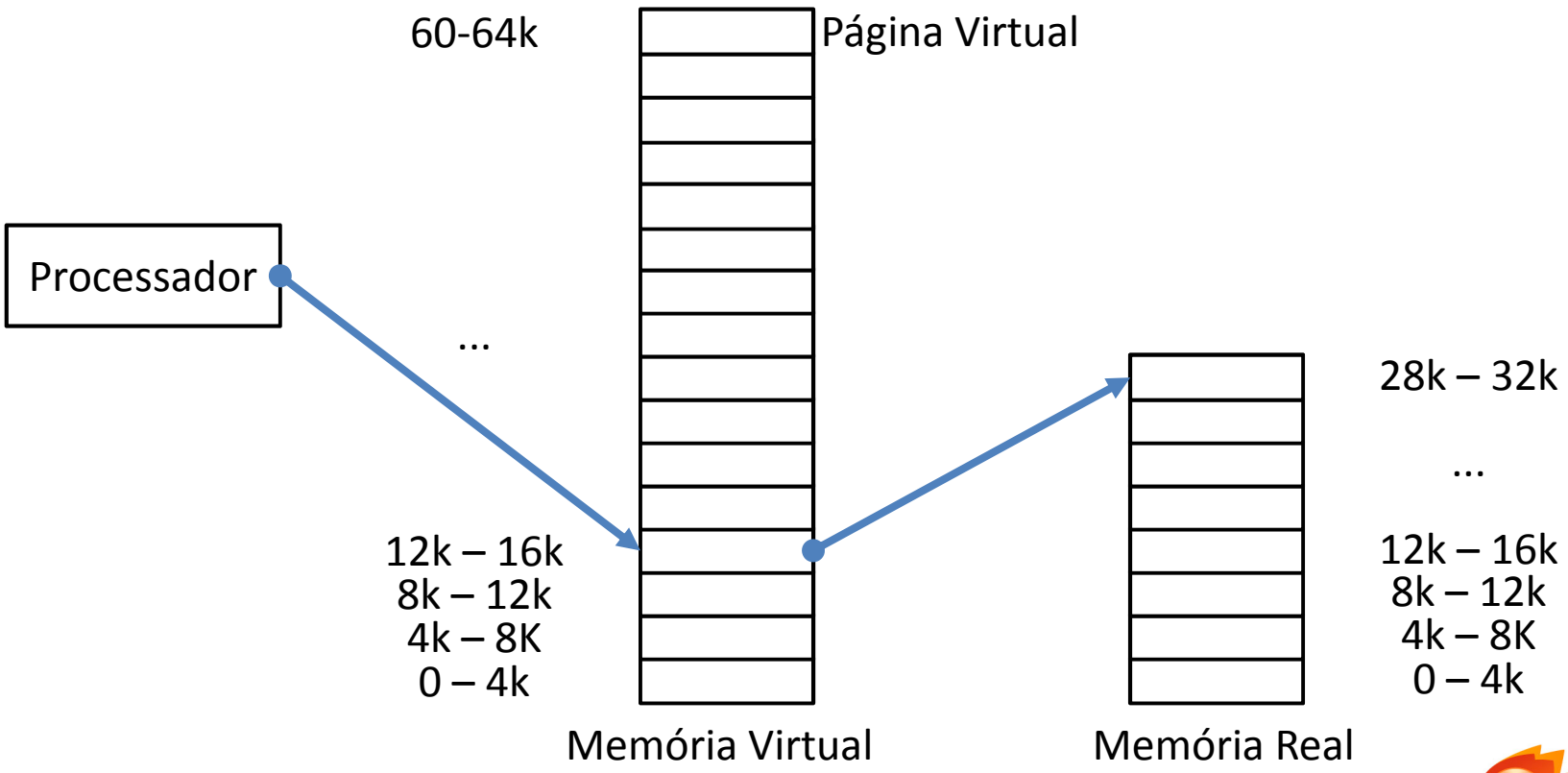


Memória Virtual - Motivação

- Programas são muito maiores que a memória principal
- Como é possível alocá-los?
  - Primeira idéia foi quebrar o programa em partes para que estas pudessem ser executadas separadamente
  - Programas ficam parcialmente na memória principal
  - O SO se responsabilizava pelo carregamento das partes
  - Programador dividia as partes (overlays)
    - Chato, complexo e demorado

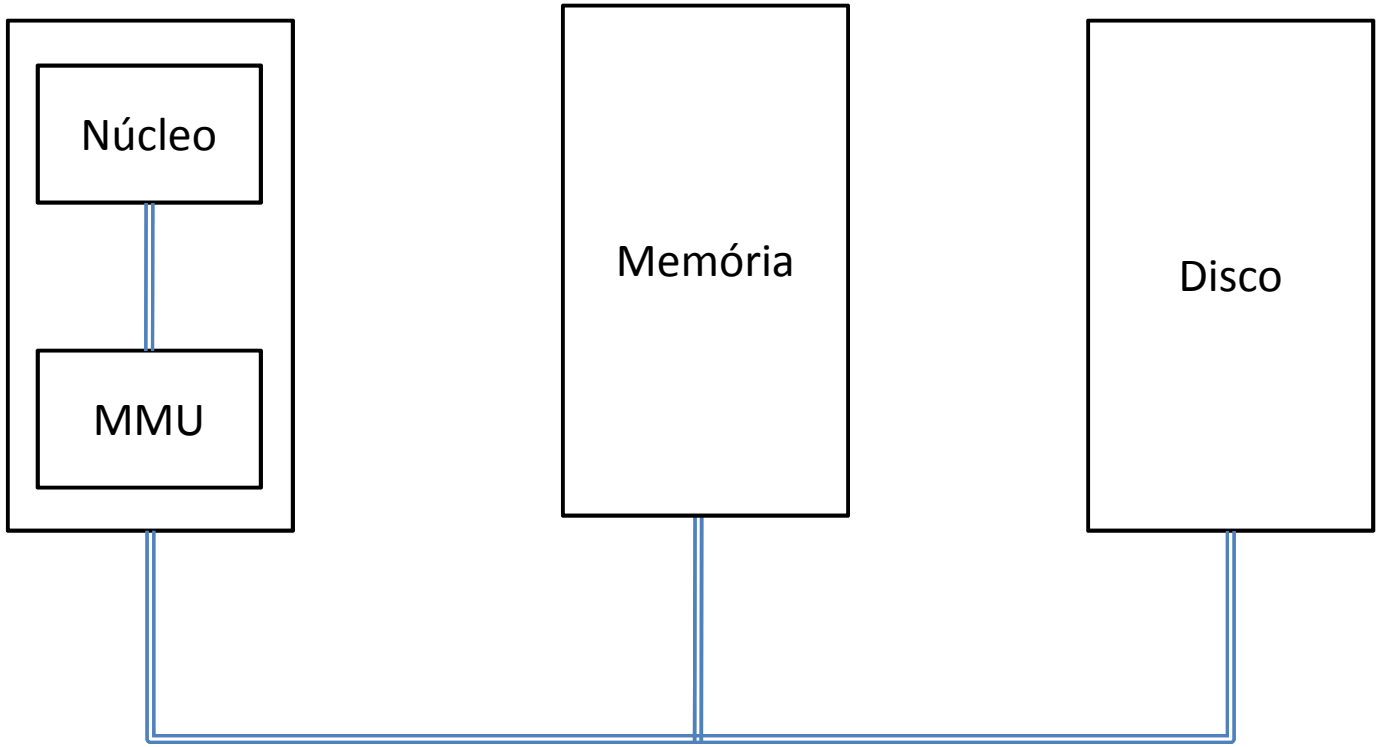
Memória Virtual

- Endereçamento virtual da memória principal
- Mas quem faz este mapeamento?



Memória Virtual

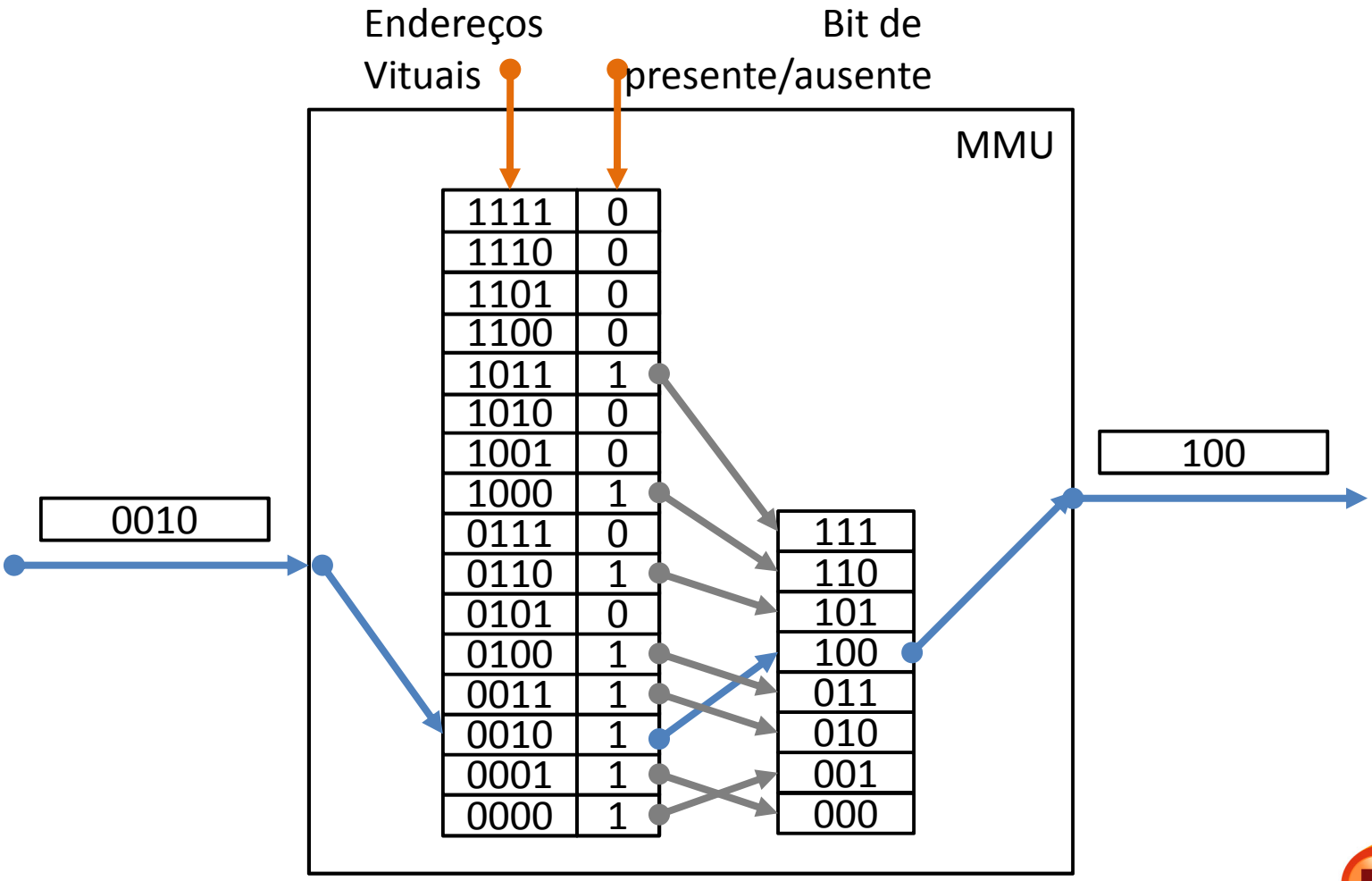
- Unidade de Gerenciamento de Memória (MMU)
  - Recebe endereços virtuais
  - Solicta endereços físicos, via barramento



Memória Virtual – Paginação

- Para a MMU localizar a página virtual e real é necessário fazer uma relação entre os endereços
- Mas a memória virtual tende a ser maior que a real
  - Como fazer o mapeamento?

Memória Virtual – Paginação

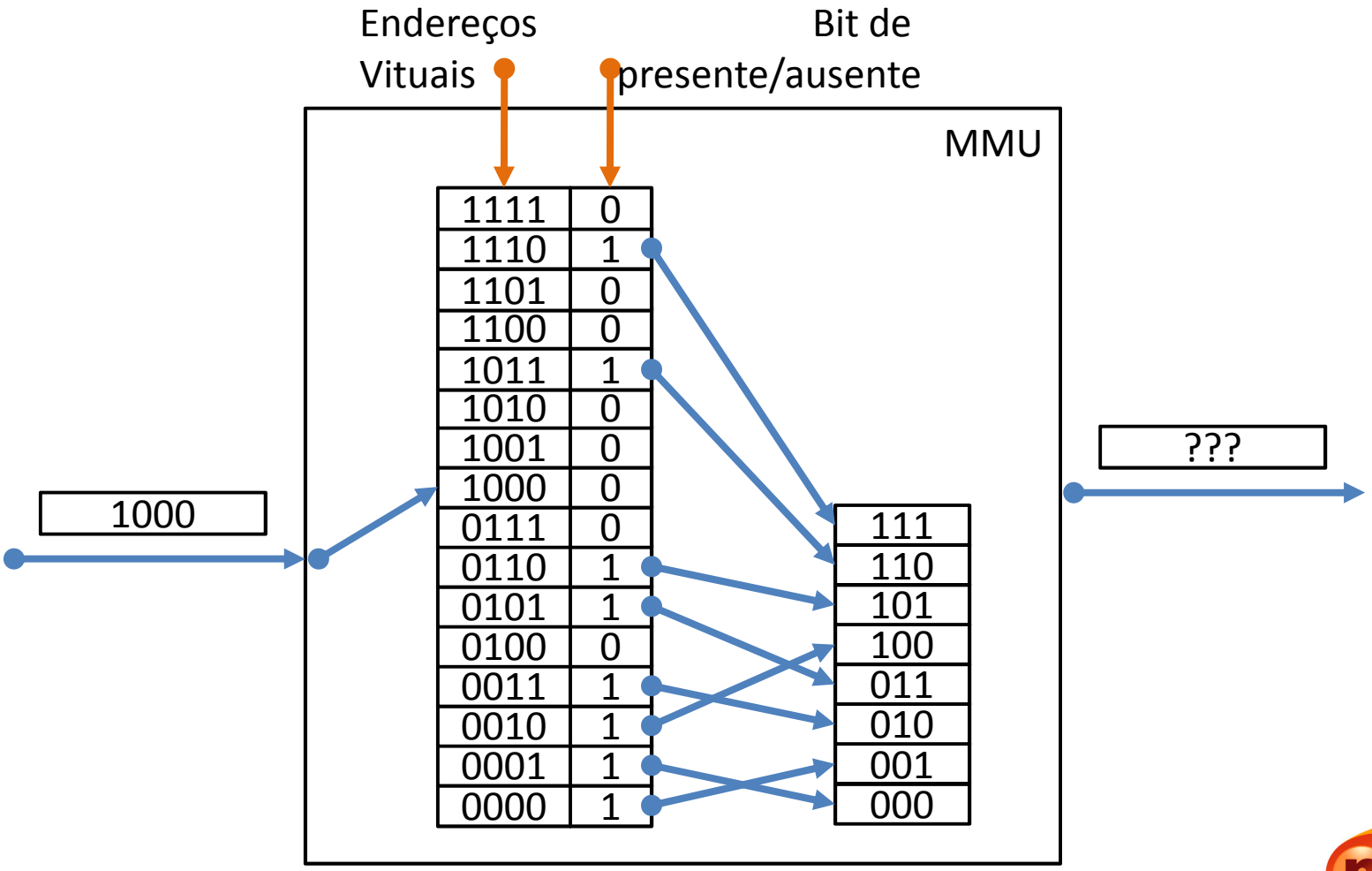


Memória Virtual – Paginação

- Para aumentar a performance MMU na tradução de endereços foi criada uma memória associativa dentro da CPU denominada TLB (*translation lookaside buffer*)

Memória Virtual – Paginação

- O que acontece se o processador pedir a página 8 (1000)?



Memória Virtual – Paginação

- Acontece uma falta de página
- A MMU deve liberar uma página da memória e resgatar a solicitada no disco
  - Mas qual das Páginas deve ser escolhida?
  - Qual o critério que deve ser utilizado?
- Algoritmo de substituição de Páginas (Ah....)

Memória Virtual – Paginação

- Como fazer um algoritmo de substituição de Páginas perfeito?
  - Fácil de pensar
  - Basta liberar a que será utilizada por último
  - Mas como definir isso?
  - Solução é retirar da memória a página referente ao processo menos utilizado
    - Mas se o usuário passar a utilizá-lo?
    - É impossível prever

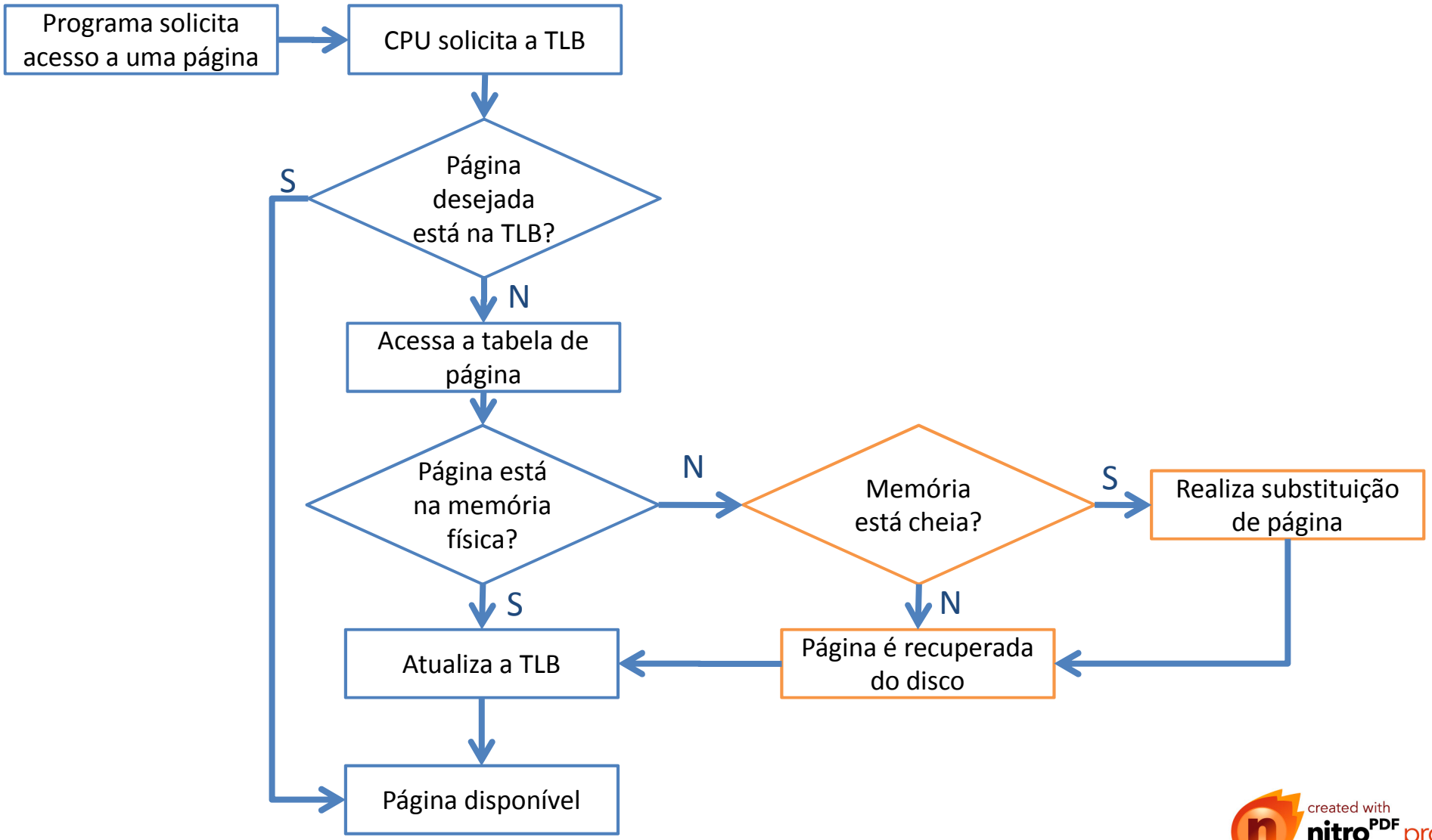
Memória Virtual – Paginação

- Como fazer um algoritmo de substituição de Páginas perfeito?
  - Fácil de pensar
  - Basta liberar a que será utilizada por último
  - Mas como definir isso?
  - Solução é retirar da memória a página referente ao processo menos utilizado
    - Mas se o usuário passar a utilizá-lo?
    - É impossível prever
- Algoritmo de Substituição de Páginas Ótimo

Memória Virtual – Paginação

- Algoritmos de Substituição de Páginas
  - Ótimo
  - FIFO
  - Segunda Chance
  - NUR
  - Relógio
  - MRU
  - Conjunto de Trabalho
  - WSCLOCK

Algoritmo de Substituição de Páginas



Algoritmo de Substituição de Páginas

- NUR (Não Usada Recentemente)
  - Idéia é simples
  - Classifica as Páginas da seguinte forma
    - Classe 0: não acessada, não modificada
    - Classe 1: não acessada, modificada
    - Classe 2: acessada, não modificada
    - Classe 3: acessada, modificada
    - Qual a mais importante?

Algoritmo de Substituição de Páginas

- NUR (Não Usada Recentemente)
  - Implementação

Falta de página

PAG	R	M
111	1	0
110	0	1
101	1	1
100	1	1
011	1	0
010	0	0
001	1	0
000	1	1

Falta de página

PAG	R	M
111	1	0
110	0	1
101	1	1
100	1	1
011	1	0
010	1	1
001	1	0
000	1	1

Falta de página

PAG	R	M
111	1	0
110	1	1
101	1	1
100	1	1
011	1	0
010	1	1
001	1	0
000	1	1

Qual escolher?

O que acontecerá depois de mais 2 falhas?

Algoritmo de Substituição de Páginas

- NUR (Não Usada Recentemente)
  - Implementação

Falta de página

PAG	R	M
111	1	0
110	0	1
101	1	1
100	1	1
011	1	0
010	0	0
001	1	0
000	1	1

Falta de página

PAG	R	M
111	1	0
110	0	1
101	1	1
100	1	1
011	1	0
010	1	1
001	1	0
000	1	1

Falta de página

PAG	R	M
111	1	0
110	1	1
101	1	1
100	1	1
011	1	0
010	1	1
001	1	0
000	1	1

Qual escolher?

O que acontecerá depois de mais 2 falhas?

Algoritmo de Substituição de Páginas

- FIFO
  - Primeira a entrar é a primeira a sair
  - Exemplo do mercado
    - Vender primeiro os produtos que compramos primeiro (validade menor)
    - Mas não podemos prever qual deles o usuário vai comprar
  - Se uma página é constantemente utilizada não tem tratamento diferenciado

Algoritmo de Substituição de Páginas

- FIFO
  - Funcionamento

Falta de página

PAG
111
110
101
100
011
010
001
000

fim da fila

4	5	6	7	0	1	2	3	4

Algoritmo de Substituição de Páginas

- Segunda Chance
  - Otimização do FIFO
  - Utiliza o Bit R
  - Escolhe página do ínicio da fila
    - Se R=0 (página não referenciada), é escolhida para sair
    - Se R=1
      - Coloca R:=0
      - Coloca no fim da fila de novo (segunda chance)

Algoritmo de Substituição de Páginas

- Segunda Chance
  - Funcionamento

Falta de página

Página escolhida!!!

PAG	R
111	0
110	0
101	1
100	1
011	1
010	1
001	0
000	1

PAG	R
111	0
110	0
101	1
100	0
011	1
010	1
001	0
000	0

PAG	R
111	0
110	0
101	0
100	0
011	1
010	1
001	0
000	0

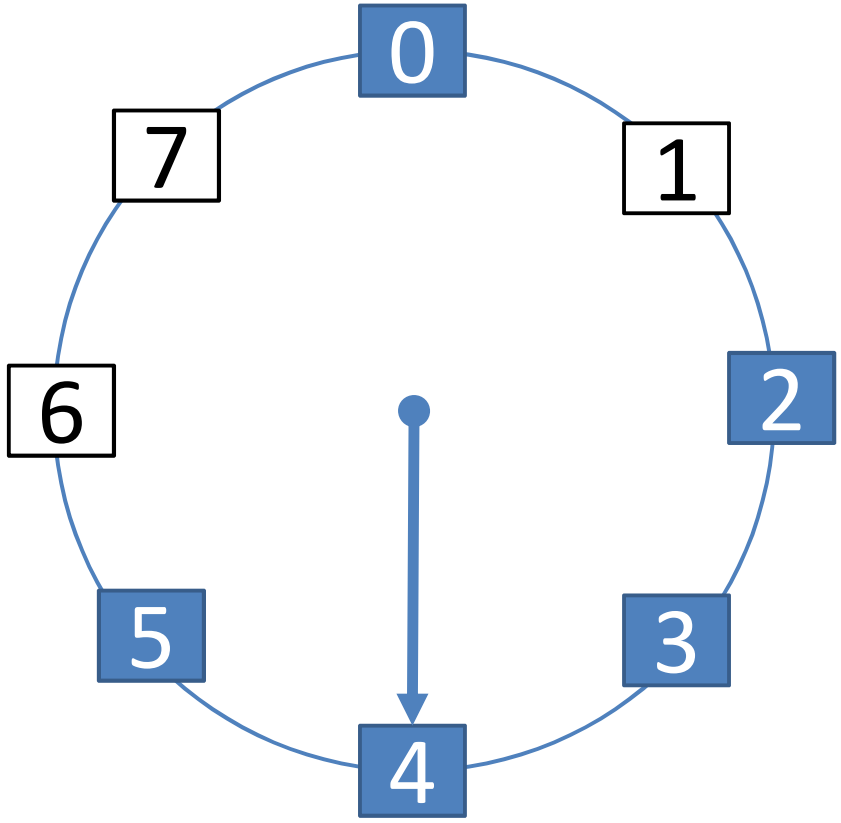
fim da fila

4 5 6 7 0 1 2 3 4 5 8

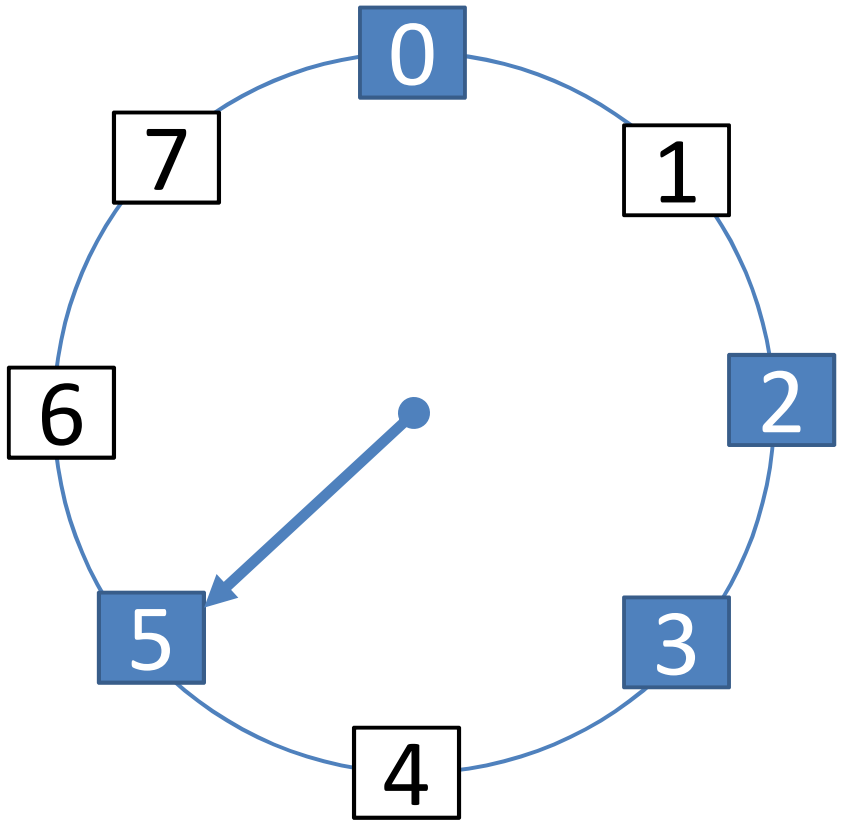
Algoritmo de Substituição de Páginas

- Relógio
  - Segunda Chance faz muitas inseções/remoções na fila
  - Relógio é uma lista circular, onde o ponteiro indica a cabeça da lista

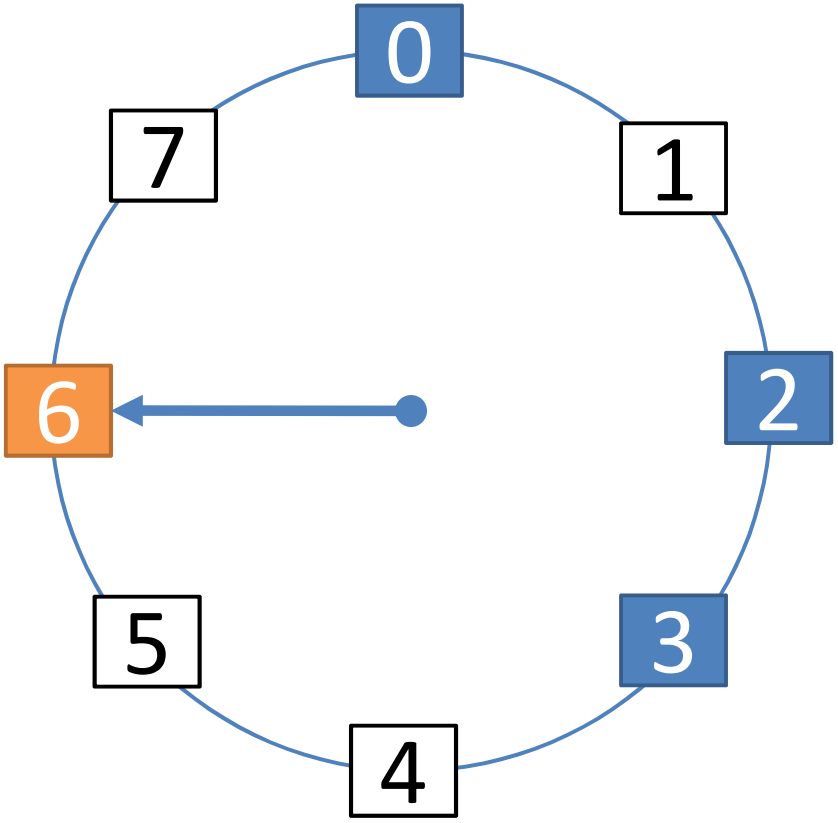
Algorimos de Substituição de Páginas



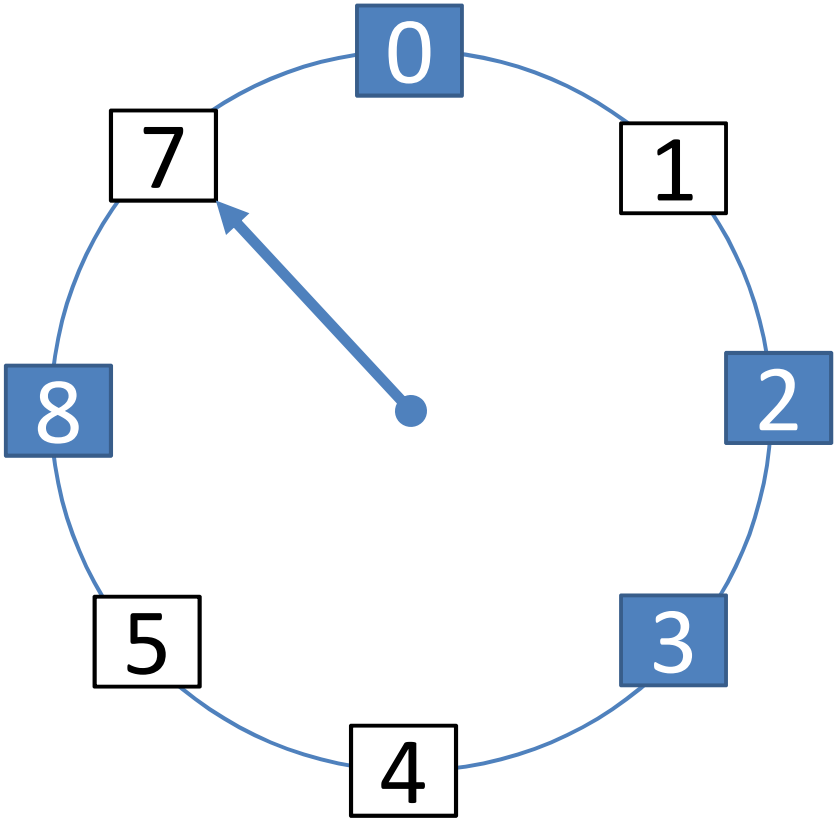
Algorimos de Substituição de Páginas



Algorimos de Substituição de Páginas



Algorimos de Substituição de Páginas



Algoritmo de Substituição de Páginas

- MRU – Menos Recentemente Usada
  - Tem um desempenho perto do ótimo
  - Idéia é : Se uma página não foi utilizada pelas últimas instruções, tem probabilidade menor de ser referenciada
  - Difícil de ser implementado (hardware)
    - Idéia com matrizes
  - Duas implementações práticas em software
    - NUF (não utilizada freqüentemente)
    - Aging (envelhecimento)

Algoritmo de Substituição de Páginas

- MRU em hardware especial
  - Implementação
    - Quando uma página é referenciada
      - Coloca 1 em todas as linhas da página X
      - Coloca 0 em todas as linhas da página X
    - Se ocorrer uma falta de página, escolhe a página cuja a soma da linha é menor

	0	1	2	3
0	0	0	0	0
1	0	0	0	0
2	0	0	0	0
3	0	0	0	0

Algoritmo de Substituição de Páginas

- MRU em hardware especial
  - Exemplo acesso a Páginas 2, 1, 2, 3
  - Quero que continuem para 0, 2, 1

	0	1	2	3
0	0	0	0	0
1	0	0	0	0
2	1	1	0	1
3	0	0	0	0

	0	1	2	3
0	0	0	0	0
1	1	0	1	1
2	1	0	0	1
3	0	0	0	0

	0	1	2	3
0	0	0	0	0
1	1	0	0	1
2	1	1	0	1
3	0	0	0	0

	0	1	2	3
0	0	0	0	0
1	1	0	0	0
2	1	1	0	0
3	1	1	1	0

	0	1	2	3
0	0	1	1	1
1	0	0	0	0
2	0	1	0	0
3	0	1	1	0

	0	1	2	3
0	0	1	0	1
1	0	0	0	0
2	1	1	0	1
3	0	1	0	0

	0	1	2	3
0	0	0	0	1
1	1	0	1	1
2	1	0	0	1
3	0	0	0	0

Já que entenderam, Qual o problema desta implementação?

Algoritmo de Substituição de Páginas

- NUF
  - Implementação
    - Para cada página é colocado um contador
    - O contador é incrementado a cada acesso
    - Quando uma falha ocorre, o programa escolhe o contador de menor valor
    - 2,1,2,3,0,2,1,0
    - Qual o problema deste algoritmo?

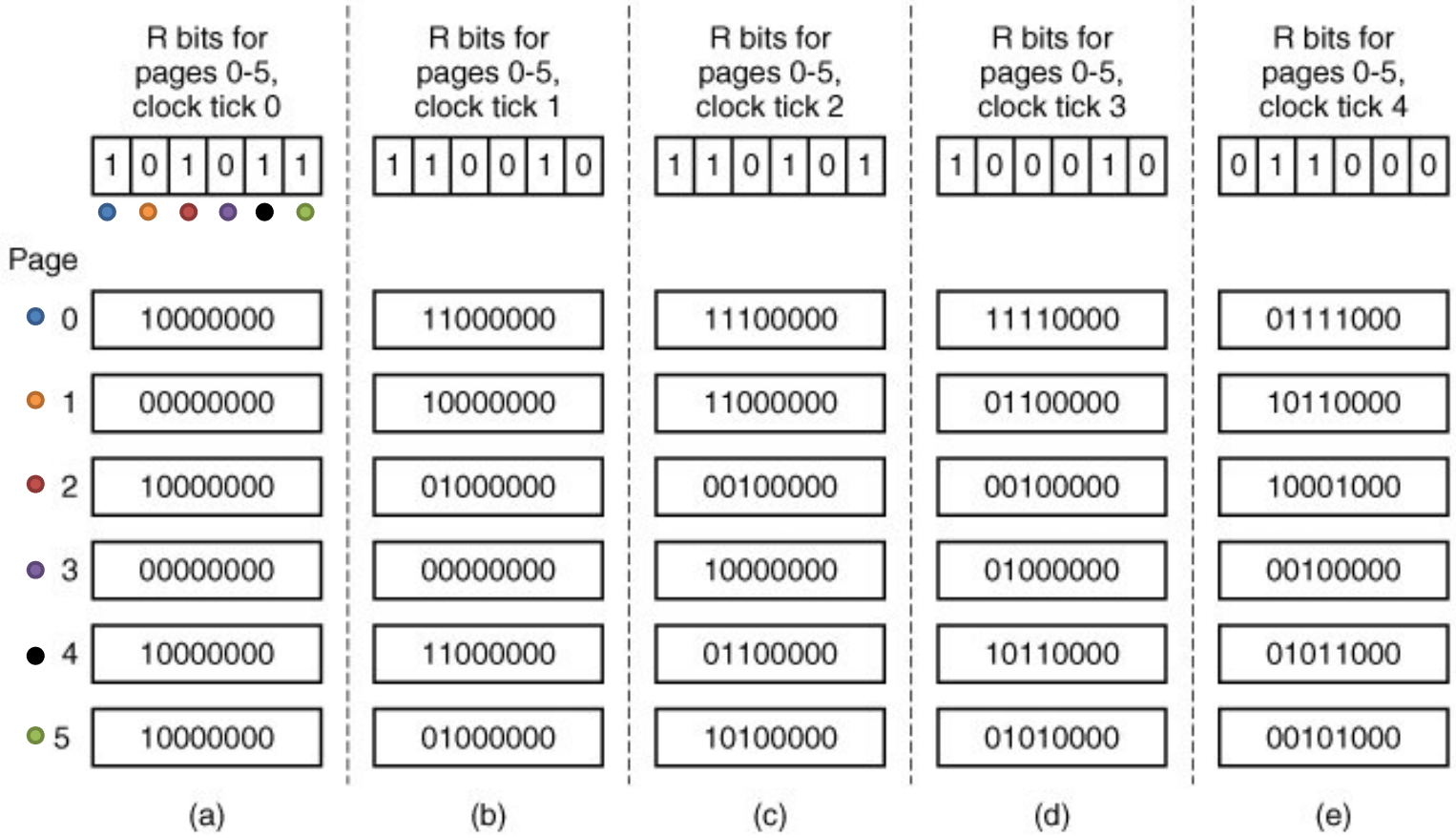
Página	R
0	2
1	2
2	3
3	1

Algoritmo de Substituição de Páginas

- Aging (envelhecimento)
  - Para corrigir o problema da persistência na escolha das Páginas mais usadas
  - Idéia é monitorar o envelhecimento das Páginas não utilizadas recentemente

Algoritmo de Substituição de Páginas

- Aging



Algoritmo de Substituição de Páginas

- Aging (envelhecimento)
  - Na prática a quantidade de bits é escolhida de acordo com o tempo de cada quantum
  - Por exemplo, se o quantum for de 25ms, uma página não referenciada nos últimos 200ms merece realmente sair

Algoritmo de Substituição de Páginas

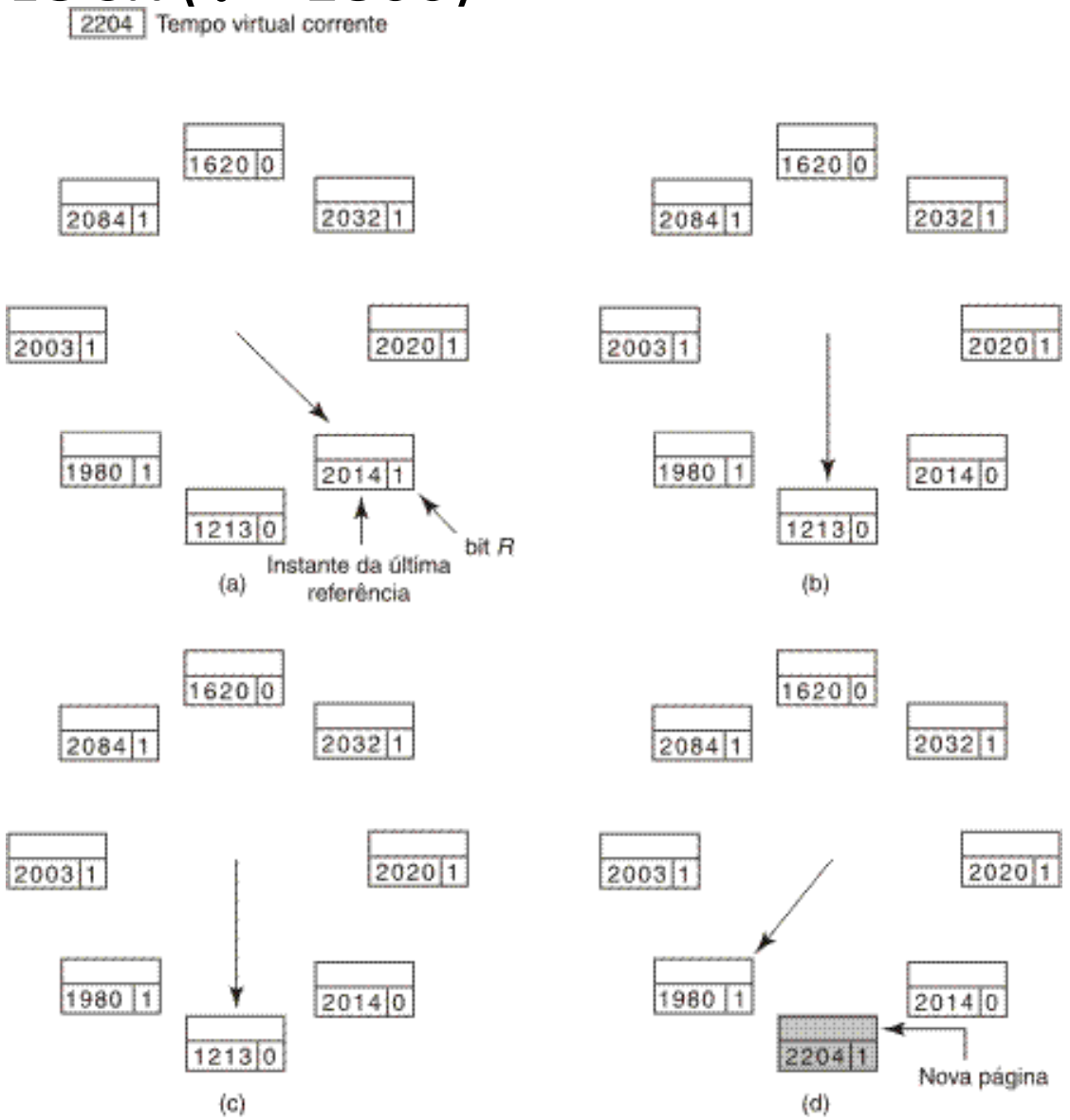
- Conjunto de Trabalho (working set)
  - Paginação por demanda
  - Todas as Páginas são carregadas independente se serão utilizadas recentemente ao não
  - Problema é que na troca de processos ocorre centenas ou milhares de falta de página de uma vez
  - Nós humanos pensamos assim... (ex. do bolo)
  - Mas é uma implementação custosa

Algoritmo de Substituição de Páginas

- WSCLOCK
  - Relógio+Conjunto de Trabalho
  - Eficiente
  - Amplamente utilizado
  - Utiliza
    - Guarda um tempo de referência
    - Lista de Páginas formando um anel a cada página carregada na memória
    - Utiliza o bit R e o tempo da última vez que a página foi referenciada;

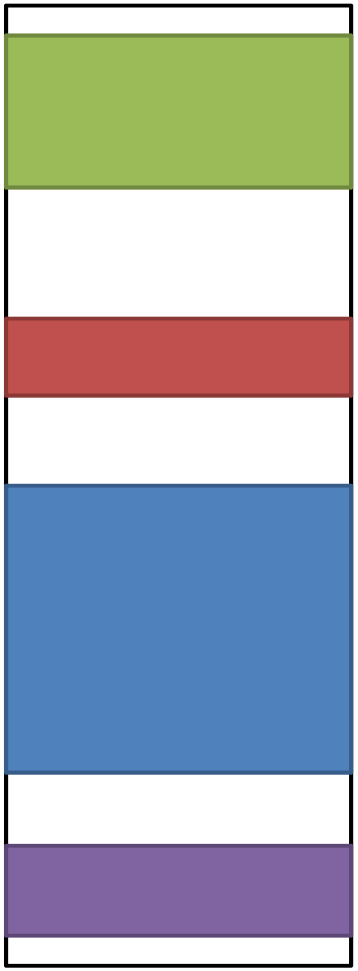
Algoritmo de Substituição de Páginas

- WSCLOCK (τ = 1800)



Conceito de Segmentação

```
program  sistemasoperacionais;

var
    a: array....
    b: ....


function x
  {comandos}

begin




end;
```



Conceito de Segmentação

```
program  sistemasoperacionais;

var
   a: array....
   b: ....


function x
  {comandos}

begin




end;
```

Segmentação

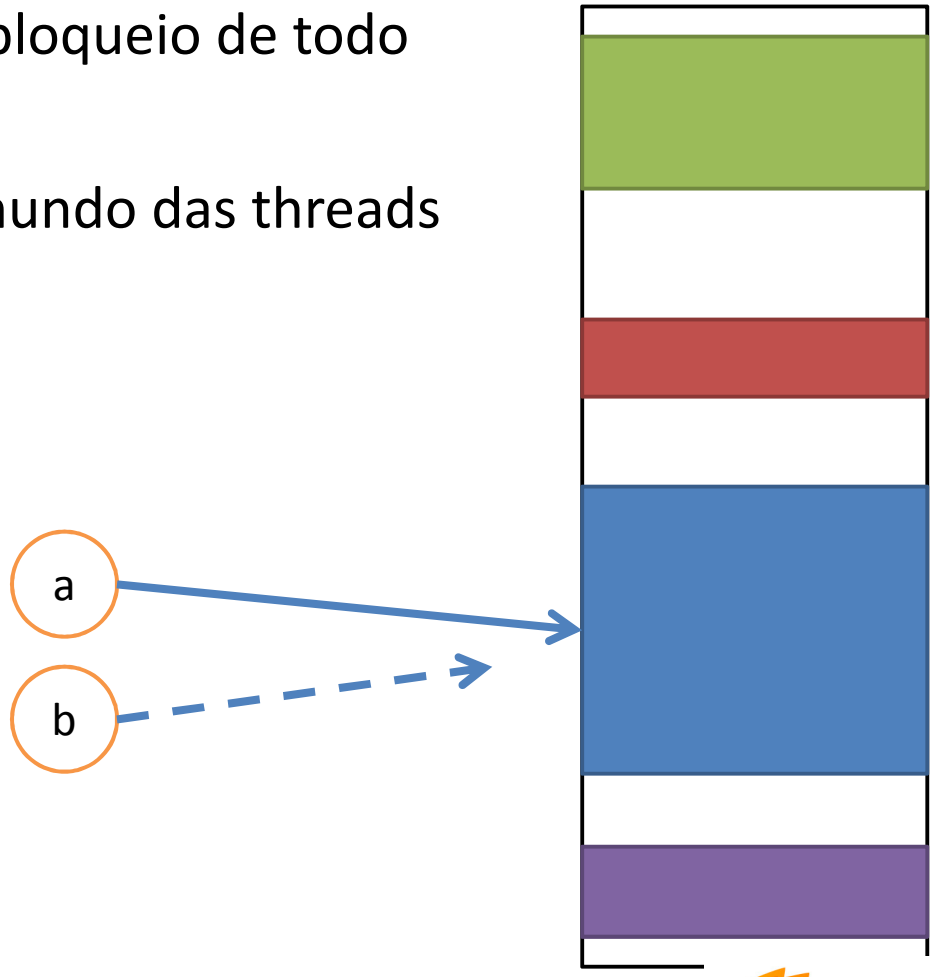
- Leva em consideração a visão de programadores e compiladores
- Um programa é uma coleção de segmentos, tipicamente:
  - Código
  - Dados alocados estaticamente
  - Dados alocados dinamicamente

Segmentação

- Leva em consideração a visão de programadores e compiladores
- Um programa é uma coleção de segmentos, tipicamente:
  - Código
  - Dados alocados estaticamente
  - Dados alocados dinamicamente

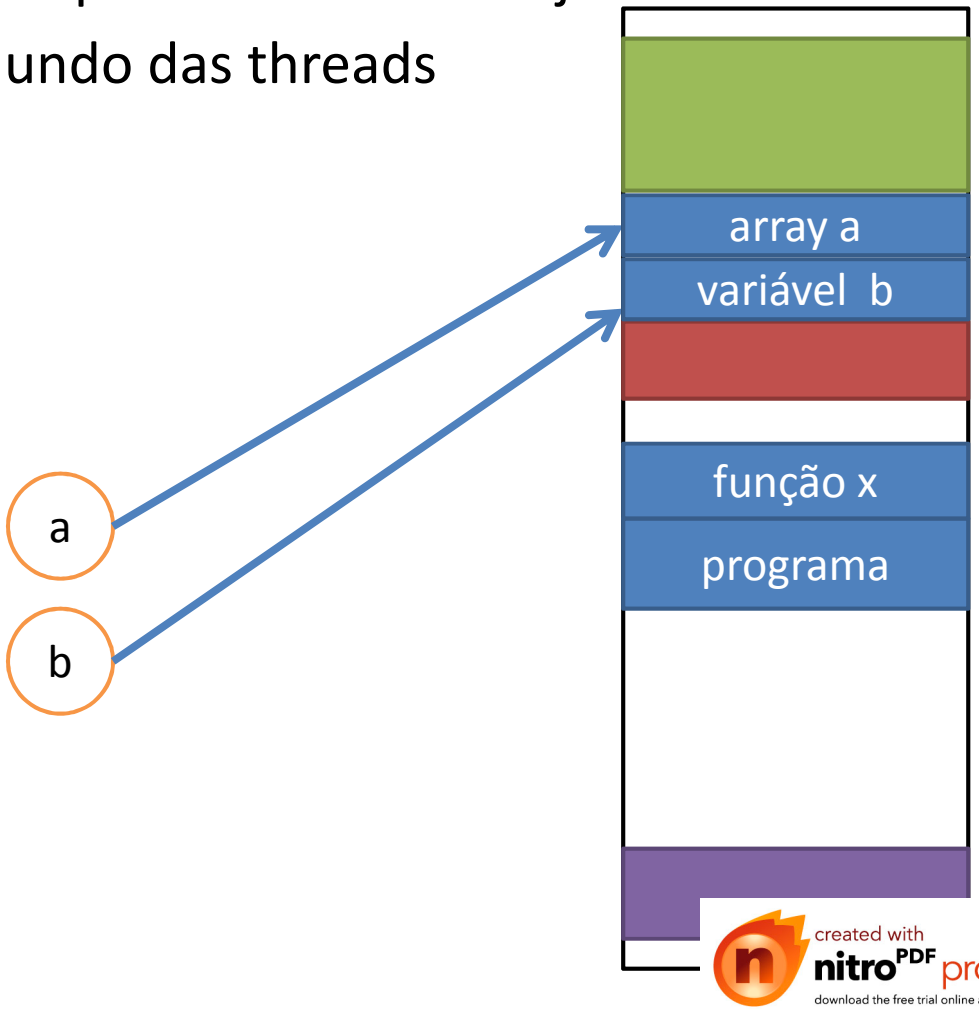
Segmentação

- Vantagem
  - Segmentos de uso comum são melhor compartilhados
    - Antes era necessário o bloqueio de todo o conjunto
    - Vantagem enorme no mundo das threads



Segmentação

- Vantagem
  - Segmentos de uso comum são melhor compartilhados
    - Antes era necessário o bloqueio de todo o conjunto
    - Vantagem enorme no mundo das threads



Segmentação

- Vantagem
  - Resolve o problema da fragmentação interna
    - Antes era necessário reservar um espaço entre código e dados para que ambos convivessem harmonicamente
    - Agora como eles são alocados separadamente, ficam independentes

Segmentação

- Em contrapartida
  - Gera fragmentação externa
    - Com mais segmentos a chance de aparecerem vários espaços vazios é maior....
    - Como solucionar?

Enquanto vocês pensam...

- Livro Arquitetura de Sistemas Operacionais

Característica	Paginação	Segmentação
Tamanho dos blocos de mémoria	Iguais	Diferentes
Proteção	Complexa	Mais simples
Compartilhamento	Complexo	Mais simples
Estruturas de dados dinâmicas	Complexo	Mais simples
Fragmentação interna	Pode existir	Não existe
Fragmentação externa	Não existe	Pode existir
Programação modular	Dispensável	Indispensável

Enquanto vocês pensam...

- Livro Sistemas Operacionais Modernos

Consideração	Paginação	Segmentação
O programador precisa estar ciente de que essa técnica está sendo usada?	Não	Sim
Quantos espaços de endereçamentos lineares existem?	Um	Muitos
O espaço de endereçamento total pode exceder o tamanho da memória física?	Sim	Sim
Os procedimentos e os dados podem ser diferenciados e protegidos separadamente?	Não	Sim
As tabelas com tamanhos variáveis podem ser acomodadas facilmente?	Não	Sim
O compartilhamento de procedimentos entre usuários é facilitado?	Não	Sim
Por que essa técnica foi inventada?	Para fornecer um grande espaço de endereçamento linear sem a necessidade de comprar mais memória física	Para permitir que programas e dados sejam quebrados em espaços de endereçamento logicamente independentes e para auxiliar o compartilhamento e a proteção

Segmentação com Paginação

- Agrega as vantagens de cada um e melhora a fragmentação interna e externa
- Permite melhor compartilhamento dos dados
- Solução é paginar os segmentos de memória
- Implementado comercialmente desde 1985
  - 80386
    - Fabricado de 1985-1994
    - 32bits da INTEL
    - Considerado um “divisor de águas” no mercado dos processadores