PCS 3216
Sistemas de Programação

Aula 14

Ligação e Ligadores. Overlays

LIGAÇÃO E LIGADORES

Introdução

- Nesta aula o foco do estudo é dirigido aos softwares responsáveis por viabilizar o desenvolvimento **modular** dos programas, em um sistema de programação.
- Como vimos na aula anterior, a modularização é obtida quando se dispõe de algum mecanismo através do qual programas independentemente desenvolvidos possam **referenciar-se mutuamente** pela citação dos **nomes simbólicos** declarados nos diversos módulos.
- Vimos ainda que os **ligadores** são os responsáveis pela resolução dessas referências simbólicas entre os módulos, e que tal operação se chama **ligação**.

- A operação de ***ligação***, que efetua a associação de endereços aos nomes simbólicos utilizados na comunicação entre módulos, é também conhecida na literatura como ***linking***, ***binding***.
- Os programas de sistema utilizados para realizar esta tarefa denominam-se ***ligadores, linkers***, ***binders***.
- Esses programas assumem diversas formas, conforme a aplicação a que se destinam e os métodos empregados para a sua realização.

Ligadores-relocadores

- Podem apresentar-se como programas muito simples, encarregados apenas de **ligar** os módulos entre si, e **relocá-los**, obtendo diretamente um programa absoluto executável.
- Conforme o caso, incorporam também as funções do **loader**, carregando diretamente na memória para execução o programa absoluto produzido.
- Tantas variantes dos ***ligadores-relocadores*** recebem também nomes diversos: ***relocadores-ligadores, relocating loaders, linking loaders***.

- As diversas funcionalidades citadas podem realizar-se através de programas que operam separadamente:
  - um ***ligador*** (***linker, binder***), encarregado apenas da resolução dos endereços simbólicos entre módulos, produzindo um programa objeto relocável sem referências simbólicas.
  - um ***alocador de memória*** (***locator***), encarregado de distribuir, na memória disponível, os módulos da melhor maneira, para efeito de associação de endereços absolutos aos mesmos. Em geral esta função é do sistema operacional.
  - um ***relocador*** (***relocating loader***), encarregado da resolução do endereçamento relativo apresentado pelas instruções de referência à memória, especialmente aquelas entre módulos, produzindo um programa objeto absoluto.
  - Um ***carregador*** **(*loader*),** encarregado de carregar na memória para execução o programa absoluto produzido pelo relocador.

- Neste esquema, e possível a operação parcial do processo completo de preparação de um programa para a execução, permitindo, independentemente:
  - a **ligação de alguns módulos** entre si,
  - a **relocação de alguns módulos** e ainda
  - a **geração de código absoluto** para alguns módulos.
- Isto muito flexibiliza o sistema, razão que justifica a adoção deste método em diversos sistemas de desenvolvimento modernos.

Overlays

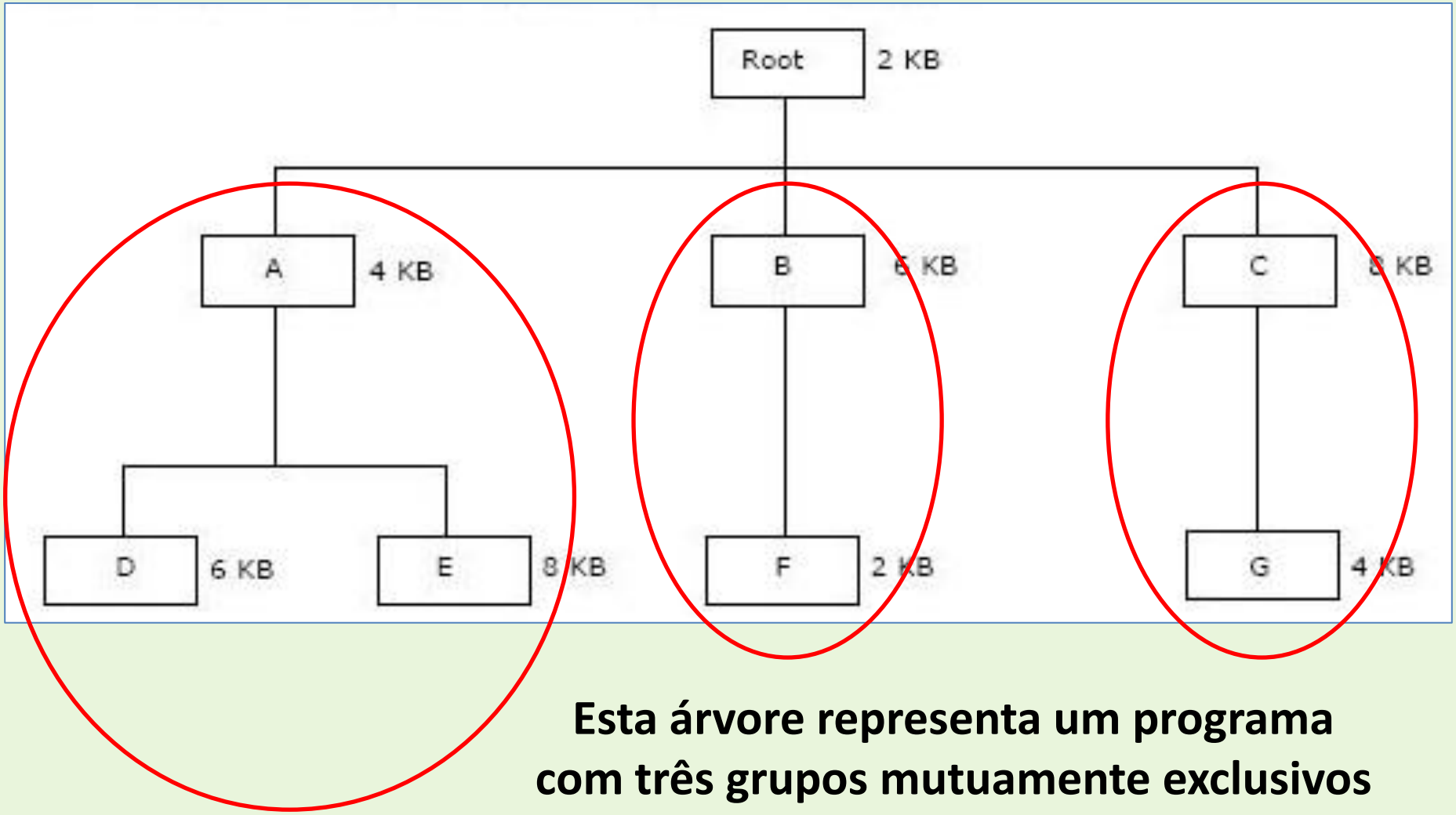
- Para alguns outros sistemas, pode ser utilizada a estratégia conhecida como **sistema de *overlays***:
- Trata-se de uma antiga mas eficaz **técnica manual** de alocação, que viabiliza executar programas grandes, mesmo com disponibilidade menor de memória física.
  - uma **área de memória** é reservada para **alojar áreas** de dados e de programas **que sejam comuns a vários módulos**
  - uma **outra (que se chama área de *overlay*)** é preferencialmente destinada a abrigar **partes mutuamente exclusivas** do programa, ou seja, aquelas das quais nunca é necessária a presença física simultânea na memória durante a execução do programa.
  - Dessa forma, partes mutuamente exclusivas do programa nunca convivem, liberando assim o máximo de área na memória física para que possa ser ocupada por outras partes que, de outra maneira, não poderiam ser executadas por falta de espaço.

Observações sobre *Overlays*

- A clássica técnica do ***Overlay*** tem como meta **permitir a execução de programas maiores que a memória** disponível.
- É do **programador** a responsabilidade de **particionar o seu programa** adequadamente, de forma que sejam **minimizadas as mudanças de contexto** durante a execução.
- Em cada instante, devem permanecer **residentes os dados e as instruções necessários** à execução do programa na ocasião.
- Pode não ser imediato projetar a estrutura de *overlays* para os programas, pois **o sistema operacional dificilmente dá suporte** automático para essa técnica.

- É usual que os programas que empregam esta técnica sejam manualmente estruturados como uma **árvore de execução**.
- **Em cada instante** da execução, precisam obrigatoriamente estar residentes na memória apenas aqueles **módulos** que correspondam aos nós **dessa árvore presentes no caminho** que leva do módulo em execução até a raiz.
- Como o sistema não entra no mérito da forma como o programa é particionado, nem da dinâmica de substituição dos seus *overlays*, é **responsabilidade exclusiva do usuário** projetar, estruturar e garantir que em tempo de execução todas as condições operacionais do programa sejam devidamente respeitadas.

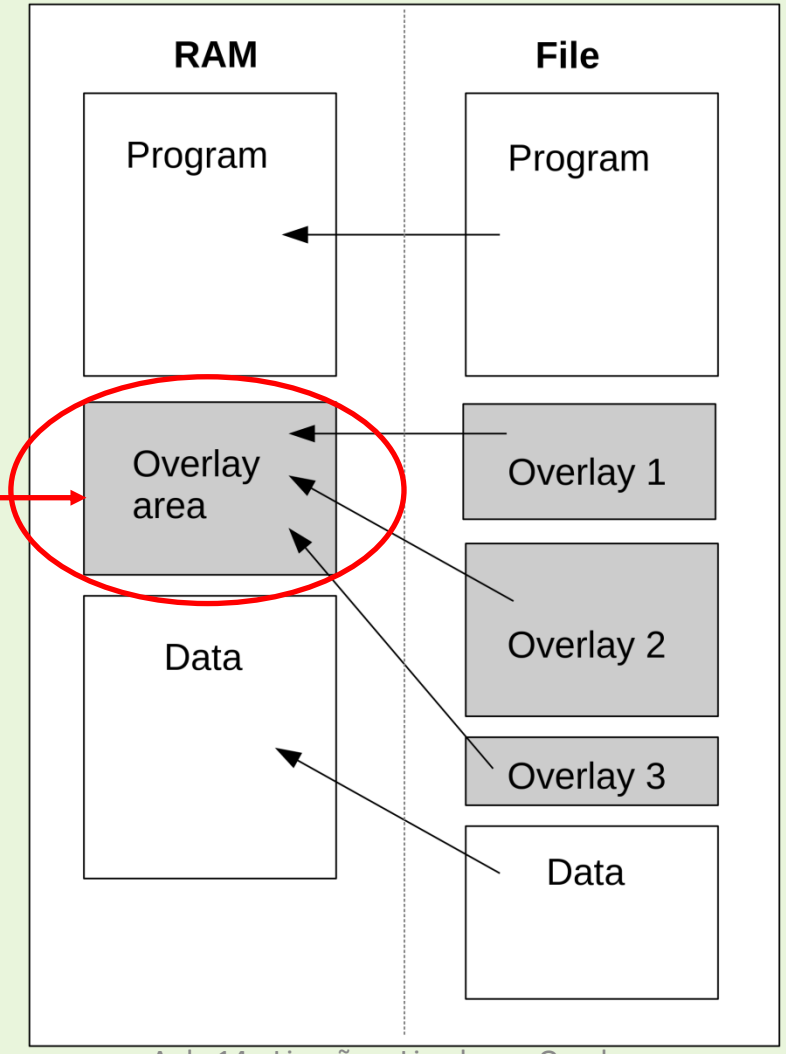
Uma árvore de overlays



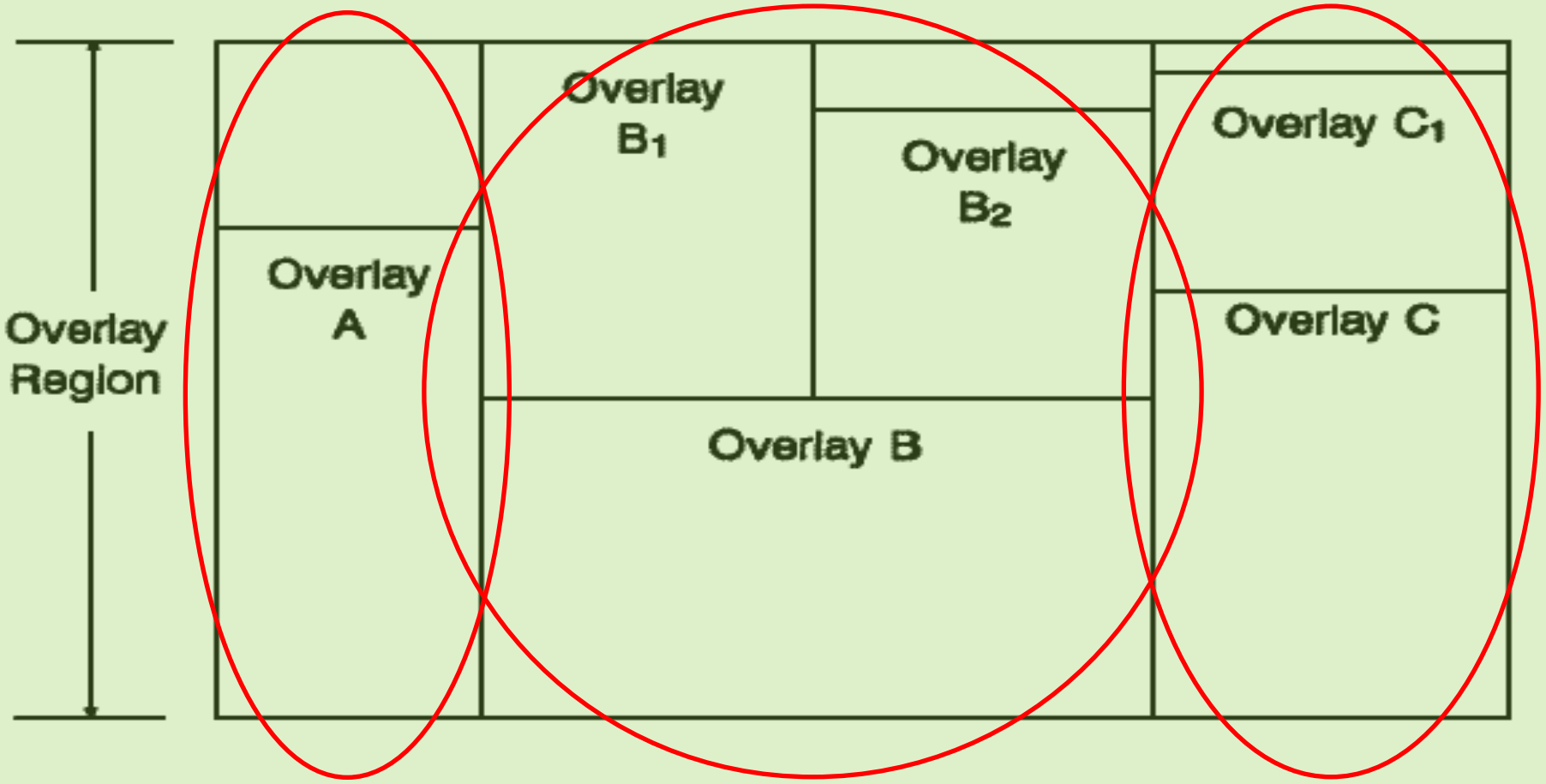
Esta árvore representa um programa com três grupos mutuamente exclusivos de módulos (delimitados em vermelho).

Operação de um programa com uma área compartilhada por 3 overlays

Os overlays 1,2 e 3 ocupam, *um de cada vez*, esta mesma área da memória RAM.



Programa com hierarquia de overlays



Os overlays A,B e C ocupam, *um de cada vez*, a mesma "Overlay Region" da memória RAM. Note que, por sua vez, B comanda sua própria área de overlay particular, na qual se revezam dois overlays menores (B_1 e B_2).

- Para o acerto e a resolução dos endereços relativos contidos nos ***overlays***, os ligadores, relocadores e alocadores deverão executar algoritmos adequados, que levem em conta as características das partes do programa que desempenhem o papel de ***overlays***.
- O uso da técnica de ***overlay*** permite ao usuário utilizar o computador para executar um programa mesmo que a área de memória disponível não seja suficiente para comportar totalmente o programa de uma só vez.

- O uso dessa técnica representa, na evolução dos sistemas de programação, um grande passo em direção aos sistemas modernos de computação.
- Convém lembrar que historicamente os ***overlays*** significaram uma decisiva vitória contra as limitações do espaço de armazenamento na memória principal.
- Os ***overlays*** acabaram representando, na história, a principal precursora da técnica da ***virtualização de memória***, amplamente empregada hoje nos sistemas modernos, e que é tema de estudo em outro contexto.

Interrupção de falta de ligação

- Alguns computadores dispõem de um suporte de hardware que permite **adiar a época da ligação** dos módulos até o instante exato em que eles venham a se fazer necessários para a execução do programa.
- Este suporte manifesta-se na disponibilidade de uma **interrupção** especial de ***falta de ligação***.
- A disponibilidade dessa interrupção na máquina permite iniciar a execução de programas mesmo que nem todos os seus módulos já se encontrem totalmente ligados e suas referências, resolvidas.

Detalhes da operação

- No instante em que é feita uma referência a um módulo ainda não ligado:
  - Ocorre uma **interrupção**.
  - O hardware provoca a execução, em modo supervisor, logo dentro do sistema operacional, de um programa de tratamento dessa interrupção.
  - Nesse programa de tratamento, **desencadeia-se a execução** de uma versão especial do programa **relocador, com a finalidade de** resolver seus endereços simbólicos e relativos.
  - Ao retornar da interrupção, o programa fica, assim, pronto para ser executado pelo hardware.
- **Notar a semelhança (intencional)** entre o hardware de interrupção e os mecanismos implementados por **um motor de eventos**.

- O esquema que acabou de ser descrito é denominado ***ligação e relocação dinâmica***.
- Uma de suas vantagens é a de possibilitar ao usuário um tempo de relocação e de ligação muito menor durante a preparação, para a execução, de programas formados de muitos módulos, quando muitos deles são solicitados raramente.
- Outra é a agilização da preparação do programa para a execução, na época de desenvolvimento de programas, devido à não obrigatoriedade da ligação e relocação completa de todos os módulos, e também dada a frequência com que as operações de ligação e relocação precisam ser feitas nessa ocasião.

- A análise pormenorizada dessas técnicas, voltadas à **virtualização de recursos** da máquina, costuma ser feito juntamente com os aprofundamentos motivados pelo estudo do funcionamento dos programas que compõem os sistemas operacionais, tema estudado em outra disciplina.

Para treinar

- Faça um estudo paralelo detalhado entre:
  - Os conceitos e processos de montagem e de ligação.
  - A lógica dos programas montador e ligador.
- Considerando o motores de evento como forma escolhida para a implementação de um montador e de um ligador:
  - Quais são os tipos de eventos no caso do montador?
  - Quais são os tipos correspondentes no caso do ligador?
  - Construa um conjunto de rotinas de tratamento que, associadas aos eventos indicados, transformam o motor de eventos respectivamente em um montador e em um ligador.
  - Esboce um programa simbólico em linguagem de montagem que defina/referencie um conjunto de no mínimo 6 e no máximo 8 símbolos.
  - Especifique três módulos que efetuem endereçamentos simbólicos ao mesmo conjunto de símbolos usados no teste do montador
  - Processe os dois testes para garantir a consistência dos dois programas