

Relatório - EP 3

Sistemas Operacionais

30 de outubro de 2014

1 Integrantes

- Victor Sanches Portella - N° USP: 7991152
- Vinícius Jorge Vendramini - N° USP: 7991103

Parte 1 - Compactação de memória

A parte 1 foi, em sua maioria, feita no arquivo `alloc.c`, na função `compacta_ep()`. Ela é chamada quando o sistema fica sem memória, o que pode ser detectado nas funções de alocação ou por first fit (levemente alterada mas nativa do minix) ou por best fit (implementada por nós na outra parte). O funcionamento dela funciona assim (em linhas gerais): percorremos a lista de buracos. Para cada buraco, vemos na lista de processos se existe algum processo imediatamente após esse buraco. Se acharmos um processo, atualizamos o buraco (movendo ele para cima e o juntando com outros buracos quando necessário) e copiamos o processo para baixo. Caso tenhamos copiado o texto do processo, atualizamos outros processos que tiverem o mesmo texto para que apontem para o lugar certo; caso tenhamos copiado os dados, basta apenas atualizar as entradas relativas ao processo movido.

Uma particularidade do Minix é a existência de um bloco no meio da memória que não corresponde a nenhum processo e que não é facilmente movido. Nós optamos por apenas pular esse bloco quando o acharmos e continuar a compactação depois dele. Além disso, nos nossos casos de teste, sempre havia o programa de teste em si rodando, o que criava um outro buraco quando ele parasse de rodar. Fora isso, a compactação funcionava para toda a memória.

Parte 2 - Política de alocação

A implementação da segunda parte está mais espalhada pelo código. Ela usa uma variável global declarada no arquivo `glo.h` do `pm` para saber se deve usar um tipo ou outro de alocação. Essa variável então é checada a cada pedido de alocação, para saber se devemos chamar a função de `first_fit` ou de `best_fit`. A chamada de sistema criada, com nome `ep_uses_best_fit(type)`, recebe 0 se não deve usar `best_fit` (e portanto deve usar `first_fit`) ou 1 se deve usar `best_fit`.

Parte 3 - Mapa de memória

A terceira parte está quase toda no `dmp_pm.c`. Ela envolve uma função que copia e ordena a tabela de processos, pega as informações restantes necessárias através de uma chamada `getsysinfo` e então imprime as informações do processo necessárias, seguidas das informações de memória. As implementações referentes à chamada `getsysinfo` e aos pedidos de informações de memória estão principalmente no `alloc.c` e no `misc.c`.

Observações

Apenas para fins de testes, criamos uma outra condição na chamada de sistema (`ep_uses_best_fit`). Como ela usava apenas valores 0 ou 1, pudemos usar quaisquer outros valores para executar uma chamada excepcional à função de compactação de memória. Assim, para testar essa função, basta chamar (por exemplo) `ep_uses_best_fit(3)`.

Os arquivos de testes estão na pasta `/usr/src/EP3-Testes`. Um deles contém uma chamada à `ep_uses_best_fit` para compactar a memória; o outro chama a `ep_uses_best_fit` com o valor passado para alternar entre `first_fit` e `best_fit`.