Algoritmo Lottery Ticket para Escalonamento de Processos

Universidade Federal de Santa Catarina
Centro Tecnológico
Departamento de Informática e Estatística
INE5412 - Sistemas Operacionais I
Artur Barichello e Lucas Zacchi
Florianópolis, dezembro de 2021

1. Introdução

Este relatório apresenta a implementação de um algoritmo de escalonamento de processos baseado em lottery tickets para o sistema operacional Nanvix..

Conforme proposto por Waldspurger[1], este algoritmo se baseia na alocação de *tickets* para os processos, de modo que seja realizado um sorteio e o processo com o ticket vencedor é escolhido para receber tempo de processador.

2. Implementação

2.1. Gerador de números pseudo-aleatórios

O grupo optou por não implementar um gerador de números pseudo-aleatórios, fazendo uso do gerador que já é implementado por padrão no Nanvix. Esse gerador é do tipo XORSHIFT e é inicializado com a seed ‘1’ que é utilizada como o estado inicial, as potenciações e deslocamentos sucessivamente acumulados geram uma sequência de números aleatória o suficiente para a aplicação. Algoritmos nesse estilo são rápidos e eficientes na geração de números porém não são seguros para aplicações criptográficas.

Foi criada uma função auxiliar nomeada *krand_range* para poder gerar números aleatórios dentro da faixa necessária.

2.2. Distribuição de tickets

Todos os processos são inicializados com zero tickets, e sempre que um processo passa para o estado *PROC_READY*, ele recebe mais 10 tickets. A alocação poderá ser aumentada através do algoritmo de Ticket Compensation onde um processo poderá acumular seus tickets e ter uma chance maior de ser selecionado.

2.3. Execução da loteria

Foi implementada a função *lottery()* que é responsável por realizar o sorteio do ticket ganhador, e encontrar o processo vencedor. Essa função é chamada a partir da função *yield()*, que atribui à *struct process *next* o processo que possui o ticket vencedor.

A função *lottery(),* por sua vez, gera um número através de um chamado à *krand_range()* com o número total de tickets no sistema para determinar o ticket vencedor. Após isso, é realizada uma verificação para encontrar o processo ganhador.

2.4. Seleção do ganhador

A seleção do processo ganhador é feita através do acúmulo dos tickets que soma sequencialmente a tabela de processos. O artigo de referência menciona um ganho de performance ao se utilizar uma lista ordenada em ordem decrescente, porém devido ao número máximo de processos do Nanvix ser pequeno foi julgado não ser necessário tal otimização.

2.5. Ticket Compensation

A compensação dos tickets é executada na função *sched()* do arquivo *sched.c*. É feito o cálculo de quanto tempo de processamento foi utilizado em relação ao tempo disponível total, a constante recebida nesse cálculo é utilizada numa multiplicação para inflar o número dos tickets disponíveis, assim aumentando a chance do processo ser selecionado novamente no futuro.

2.6. Estruturas de dados utilizadas

Na solução proposta não foi necessário adicionar novas estruturas de dados, apenas modificar as já existentes. A *struct* de processos ganhou uma variável do tipo inteiro para definir a quantidade de tickets que cada processo acumula. Foram reutilizadas as estruturas da tabela de processos para fazer a soma de tickets totais e a implementação do escalonamento do tipo loteria.

3. Testes

3.1. test sched

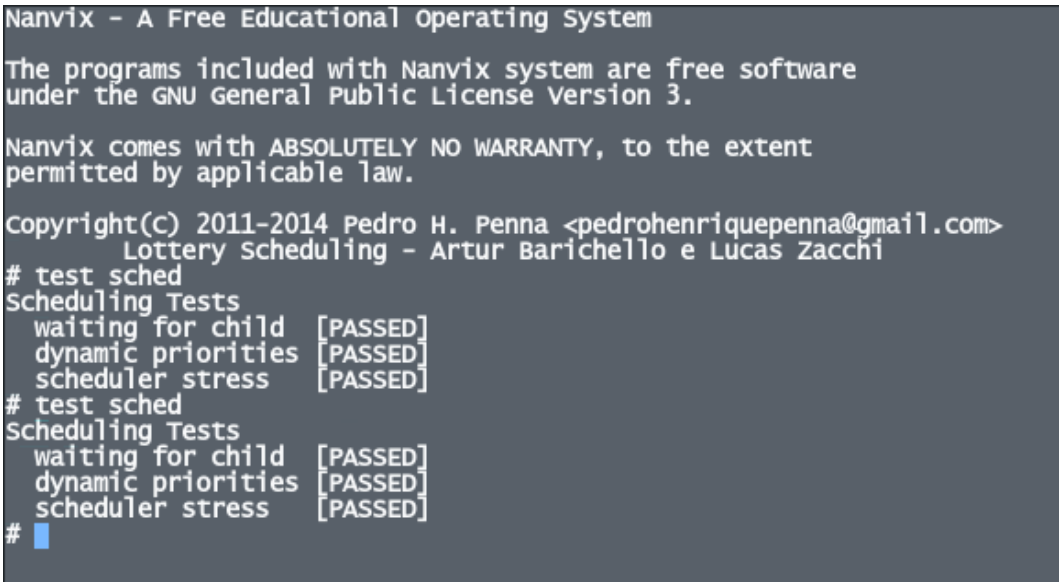

Figura 1: Execução do teste "*test sched"* do Nanvix.

4. Referências

[1]Lottery Scheduling: Flexible Proportional-Share Resource Scheduling: https://www.usenix.org/legacy/publications/library/proceedings/osdi/full_papers/waldspurger.pdf