Componentes: Naomi Ribes; Anna Gabriele

1. INTRODUÇÃO

A proposta de trabalho é a de um zoológico controlado por threads, cuja rotina diária envolve os animais, que possuem suas ações básicas de comer, mostrar-se, comer novamente e dormir, as quais são repetidas duas vezes ao dia, os veterinários responsáveis pela alimentação e um fornecedor responsável pelo reabastecimento do estoque de comidas.

Os animais são divididos em três espécies, leões, suricatos e avestruzes, cada um com uma população de tamanho diferente com seus devidos hábitos alimentares, de sono e de exibição; Seus comedouros também possuem uma quantidade limitada de espaço para a comida e o mesmo se dá para as reservas de comida de cada espécie; Fornecedores são chamados toda vez que algum estoque é esvaziado e então preenche o de todos os animais.

2. DESENVOLVIMENTO

O código foi escrito utilizando C e, principalmente, a biblioteca pthread; Ele foi organizado de forma que os animais possuem uma estrutura genérica que armazena seu total consumido, bem como flags de ação e dormir e a própria thread; Esta estrutura é então declarada globalmente em vetores com o tamanho da população de cada espécie. Outras threads como veterinário e fornecedor não armazenam nenhuma informação extra atrelada a si, portanto são apenas declarados como pthreads. Um único mutex controla o código e apenas funções básicas da biblioteca são utilizadas, sem exploração de prioridades ou políticas de fila. Há também uma estrutura de mensagem, alocada dinamicamente na função principal e responsável pelo armazenamento e transferência de informação para as threads, de forma a ter melhor controle sobre as ações dentro da rotina.

A rotina é uma função única que trata todas as threads, nela é atualizado o contador de horas, que por sua vez influencia nos ciclos de sono dos animais e no reset das suas ações que são performadas duas vezes ao dia. Todos os animais possuem sua sub rotina comer, que na realidade trata tanto a alimentação quanto as demais funções, se exibindo e iniciando o contador para dormir. Foram definidas constantes de tempo que podem ser utilizadas na função de rotina para alterar o tempo decorrido da simulação, sendo definida para uma semana por padrão. Ao fim do último dia o relatório de consumo de todos os animais é apresentado.

Instruções para compilação: gcc -pthread teste.c

3. DIFICULDADES

A maior dificuldade foi a implementação inicial, tentamos fazer por partes e construir o código mais simples e então adicionar as threads e mutex após, o que acabou não funcionando visto que a lógica para escrever um algoritmo multi thread é bastante diferente da sequência padrão de código. A partir do momento que percebemos que o avanço ficava letárgico decidimos criar um repositório novo, com o código levemente estruturado em pseudocódigo e então refeito na nova estrutura.

Tentamos também fazer o controle de tempo via biblioteca por sleep mas isso se provou problemático visto que da forma que o código foi organizado o programa acabava por congelar por completo; Nossa solução foi a criação da função de monitoração de tempo e um contador que decrementa o tempo de sono do animal.

4. CONCLUSÃO

Após a implementação fomos capazes de perceber as grandes diferenças na estrutura de programação de algoritmos multi thread e a maior necessidade de planejamento prévio, pois não é uma lógica tão intuitiva quanto a mentalidade sequencial single threaded. Para trabalhos futuros esse processo de planejamento pode ser abordado mais cedo a fim de evitar a necessidade de reconstruir o projeto inteiro. Quanto ao código final acreditamos que este não ficou tão optimizado quanto poderia, talvez um incentivo interessante seria explorar a possibilidade de melhor otimização e uso eficiente da biblioteca de threads.