Thiago Bazilio e Weberti Silva

RELATÓRIO

Trabalho apresentado Engenharia da Computação da UNIVALI - Universidade do Vale do Itajaí, para a disciplina Sistemas Operacionais.

Professor: Felipe Viel

Itajaí - Santa Catarina

Introdução

Este trabalho tem como objetivo apresentar dois projetos, cujo primeiro projeto busca mostrar uma análise comparativa de tempo de processamento utilizando bibliotecas como time.h, na qual usou-se multiplicação matricial e posicional para realizar a comparação do sistema single thread e multithread.

No segundo projeto foi feito um algoritmo de ordenação de vetores, onde o vetor é de 200 posições e foi comparado com um sistema de Singlethread e Multithread.

Projeto 1 códigos importantes e contexto da aplicação para a compreensão dos resultados obtidos.

Foi usado da linguagem c++ para o trabalho, na qual as funções utilizadas para obter os resultados da comparação de single thread e multithread foram:

Alocar Linha: Usado para alocar as linhas das matrizes.

```
//aloca linhas
int** a = new int* [linha];

int** b = new int* [linha];

int** res = new int* [linha];
```

Matricial: Utilizado na multiplicação matricial das matrizes por meio de multithreads.

```
//matricial
void matricial1() {
    for (int i = 0; i < 5000; i++) {
        for (int j = 0; j < coluna; j++) { ... }
    }
}

void matricial2() {
    for (int i = 5000; i < 10000; i++) {
        for (int j = 0; j < coluna; j++) { ... }
    }
}

void matricial3() {
    for (int i = 10000; i < 15000; i++) {
        for (int j = 0; j < coluna; j++) { ... }
    }
}

void matricial4() {
    for (int i = 15000; i < linha; i++) {
        for (int j = 0; j < coluna; j++) { ... }
    }
}
```

Posicional: Utilizado na multiplicação posicional das matrizes por multithreads.

```
//posicional
void posicional1() {
    for (int i = 0; i < 5000; i++) { ... }
}

void posicional2() {
    for (int i = 5000; i < 10000; i++) { ... }
}

void posicional3() {
    for (int i = 10000; i < 15000; i++) { ... }
}

void posicional4() {
    for (int i = 15000; i < linha; i++) { ... }
}
```

Single: Função Singlethread usado na multiplicação.

```
///Single

//matricial
void munica() {
    for (int i = 0; i < linha; i++) {
        for (int j = 0; j < coluna; j++) {
            res[i][j] = a[i][j] * b[j][i];

        }
    }
}

//posicional
void punica() {
    for (int i = 0; i < linha; i++) {
        for (int j = 0; j < coluna; j++) {
            res[i][j] = a[i][j] * b[i][j];
        }
    }
}
```

A seguir foram criadas outras 3 funções matricial para, posicional para e criar na qual serve para a criação das matrizes.

```
void matricial() { ... }

void posicional() { ... }

void criar() { ... }
```

Resultados obtidos

Simulação: Foi usado dos valores x e y para a pesquisa resultado na solução a seguir.

Tabela Comparativa: Resultados obtidos comparando Singlethread e multithread.

Conclusão do primeiro projeto

Podemos concluir que a Singlethread se saiu.... em comparação com a multithread, assim sendo, ...

Projeto 2 códigos importantes e contexto da aplicação para a compreensão dos resultados obtidos.

Resultados obtidos

Simulação: Foi usado dos valores x e y na ordenação de vetores para a pesquisa, o resultado na solução foi.

Tabela Comparativa: Resultados obtidos comparando a ordenação em Singlethread e multithread.

Conclusão do segundo projeto

Sendo assim, podemos analisar que a Singlethread se saiu de modo mais.... em comparação com a multithread que foi.....