Relatorio PCAM

- Daniel Sá Barretto Prado Garcia 103734344
- Felipe Guilermmo Santuche Moleiro 10724010
- Laura Alves de Jesus 10801180
- Tiago Marino Silva 10734748

Particionamento

Para a criação do vetor de tamanho N, ele pode ser parcionado em N tarefas, cada tarefa possui uma posição do vetor e é responsavel por setar ela como 1 ou como N (se esse elemento for o central). Após a criação do vetor, que pode ser feita em paralelo, nós devemos buscar o maior elemento no vetor, para isso podemos dividir o vetor em N tarefas em que cada tarefa verifica seu maior local(na pratica cada thread pegara mais de uma tarefa por vez), e em seguida compara com todas as outras tarefas para ver qual tem um maior global.

Comunicação

Acontece na barreira e na região crítica. Identifica que é necessário possuir o vetor pronto para encontrar o maior elemento em cada seção (barreira) e que é necessário ter o maior elemento de cada seção para achar o maior elemento do vetor (região crítica).

Aglomeração

Atribui N/T tarefas (sendo N o numero total de tarefas e T o número de threads) para cada thread. Dessa forma, antes da barreira, cada thread preenche os valores de um setor do vetor. Depois da barreira, é atribuido um setor do vetor à cada thread e é encontrado o maior elemento naquele setor. Ao chegar na região crítica, a thread atualiza uma variavel compartilhada com o maior elemento encontrado caso ele seja maior que o valor presente na variável compartilhada.

Mapeamento

As nossas threads serão escalonadas aos nucleos do processador pelo SO, a partir de uma abordagem dinamica. A ideia seria criar um numero de threads igual ao numero de nucleos do processador, a depender do número de núcleos da máquina, e desta forma esperamos que o SO escalone cada thread a cada nucleo para a melhor otimização.