1 Título do relatório

Tratabilidade por parâmetros fixos em paralelo para o problema da mochila multidimensional

2 Nome completo do orientador e do estudante

Estudante: Alison Vinicius Soares Martins

Orientador: Henrique Mongelli

3 Atividades planejadas no plano de trabalho do estudante (plano de trabalho aprovado no edital)

Etapa	2020					2021						
	A	S	O	N	D	J	F	M	A	M	J	J
Atualização bibliográfica	X	X	X									
Estudo das características das meta-heurísticas descritas		X	X	X								
Desenvolvimento de Metaheurísti-cas Híbridas				X	X	X	X	X				
Implementações e Testes							X	X	X	X	X	
Elaboração de Artigos									X	X	X	
Elaboração de Relatório Final											X	X

4 Alterações no plano de trabalho do estudante e as dificuldades encontradas

Alguns artigos estudados trouxeram dificuldade em compreender o algoritmo dado pelo autor e checagem dos resultados obtidos com os exemplos dados, o que levou a estudos mais aprofundados em algoritmos exatos para o problema da mochila 0-1 e o problema da mochila com 2 dimensões.

5 Resumo

O problema da mochila 0-1 é um problema NP-Completo, composto por uma mochila contendo uma capacidade W, uma entrada de n itens x_i cada um com um valor p_i e um peso w_i. A maximização deste problema retorna um conjunto X de itens que formam o resultado com o valor máximo. Uma extensão da mochila 0-1 é a mochila multidimensional, que possui mais de uma capacidade para a mochila e, utilizando paralelismo junto com um algoritmo exato da mochila 0-1, podemos fazer uma adaptação para o problema da mochila de 2 dimensões, assim encontramos a resposta para este problema utilizando menos espaço para o cálculo.

6 Introdução

O problema da mochila 0-1 é um problema que possui uma mochila de tamanho W e um conjunto de x_i itens, em que i = 1, ..., n, cada um com seu respectivo valor p_i e tamanho w_i. Neste problema queremos encontrar um subconjunto de itens que, quando empacotados na mochila, tenham o valor total maximizado, ou seja, o melhor valor possível para este conjunto de itens. O algoritmo para este problema leva um tempo de $O(nW)$ e um consumo de espaço $O(nW)$, mesmo tendo a aparência de um problema simples é um algoritmo de tempo pseudopolinomial. Existe outro algoritmo, chamado DP-3 (Kellerer et al., 2004), que resolve o problema da mochila no tempo $O(n^2W)$ em que n fica elevado ao quadrado pelo laço de repetição que preenche várias vezes os vetores de resposta do problema e, em alguns casos, o tempo fica $O(knW)$ com k sendo menor que n. O consumo de espaço deste algoritmo é

$O(n + W)$, tendo um vetor z com W posições usado para o cálculo e um vetor X com n posições que guarda os índices de cada item empacotado.

Existe uma extensão do problema da mochila chamado problema da mochila multidimensional, onde a capacidade da mochila não se limita a somente uma e sim de mais capacidades ou dimensões, assim temos $W = (W_0, ..., W_d)$, sendo d o número de dimensões para o problema, com isso, cada item terá o seu respectivo valor p_i e d tamanhos, ou seja, $w_0, ..., w_d$. Um algoritmo para este problema tem o tempo $O(nW_0 \cdot ... \cdot W_d)$ e consumo de espaço $O(nW_0 \cdot ... \cdot W_d)$, tornando-o impraticável pelo alto consumo de tempo e espaço. Reduzindo este problema para duas dimensões e adaptando o algoritmo DP-3, conseguimos resolver utilizando um espaço de $O(n+W_0W_1)$ e tempo $O(knW_0W_1)$ que o torna custoso quando falamos de tempo. Podemos contornar este problema utilizando o paralelismo com GPGPU e a biblioteca CUDA (NVIDIA, 2021), com isso o preenchimento da matriz z necessita apenas de W_0+W_1 execuções dentro de um laço com k repetições. Este preenchimento é rápido devido ao comportamento da condição do problema da mochila: enquanto uma thread calcula a posição de z que está vazia, as demais threads podem realizar cálculos nas posições anteriores já calculadas.

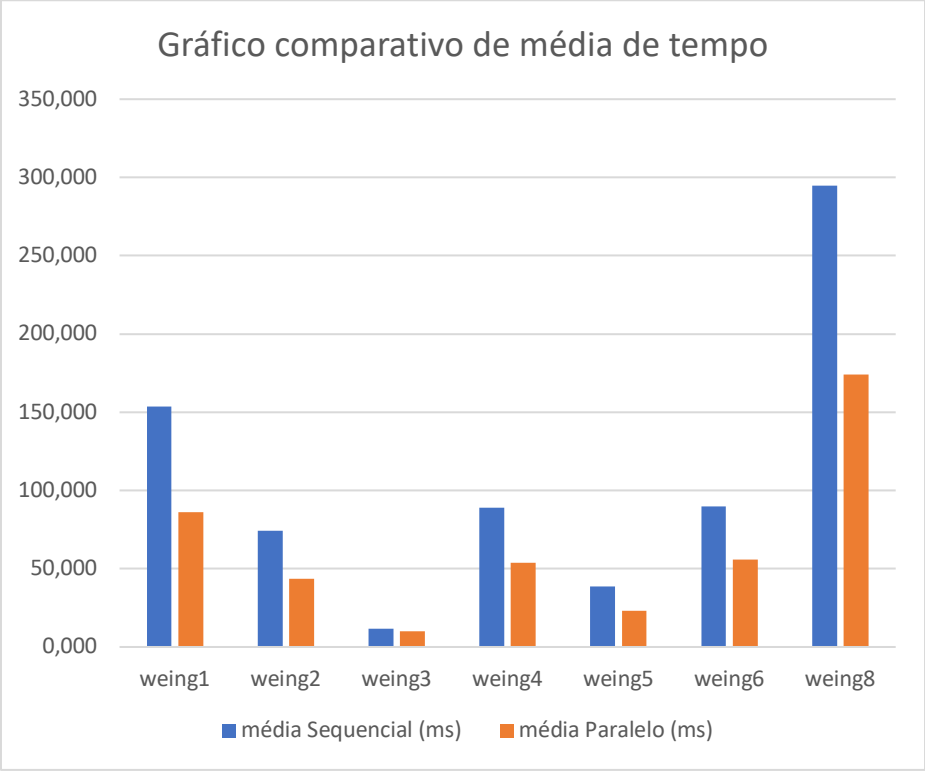
7 Metodologia

A execução deste algoritmo, tanto sequencial quanto paralela, foi realizada em uma máquina com processador intel i7 9750h com 6 núcleos e 12 threads, uma gpu GTX 1650 com 4gb de VRAM, 8gb de RAM, SSD de 512gb e sistema operacional Windows 11. Para a compilação de ambos os algoritmos foi utilizado o compilador NVCC (NVIDIA, 2021). Foi utilizado o conjunto de dados weing (WEINGARTNER, 1967) com os testes de 1 até 8, com exceção do teste weing7 que devido ao tamanho da instância e o limite de threads por bloco da GPU não foi possível encontrar o resultado exato. A média foi calculada com a soma de tempos em milissegundos de 10 execuções dividida por 10.

8 Resultados parciais /ou finais;

Na tabela encontra-se o nome dos testes executados e a média de tempo das 10 execuções de cada teste. O gráfico mostra lado a lado o tempo sequencial e paralelo de cada teste, com exceção do teste weing7.

teste	média Sequencial (ms)	média Paralelo (ms)
weing1	153,790	86,176
weing2	74,457	43,736
weing3	11,839	10,133
weing4	89,157	53,958
weing5	38,552	23,165
weing6	89,617	56,028
weing8	294,604	174,217



9 Conclusões parciais /ou finais

Com os resultados obtidos podemos ver que o paralelismo pode ser uma ferramenta útil para melhorar o tempo de execução do algoritmo. Somente com

a paralelização de uma parte do algoritmo DP-3, onde ocorre o preenchimento da matriz de cálculo *z*, foi possível uma *speedup* de 1.5x em relação ao algoritmo sequencial no tempo de execução. Muitos algoritmos podem ser combinados com heurísticas e paralelismo com GPUs, assim, fica como uma motivação para futuros trabalhos com o problema da mochila multidimensional esta combinação.

10 Referências;

WEINGARTNER, NESS; Operations Research. *Methods for the solution of Multidimensional 0/1 Knapsack Problem*, 15, 83-103, 1967.

Kellerer, N. et al. *Knapsack Problems*. Springer, Berlim, Alemanha, 2004.

NVIDIA Corporation. *CUDA C++ Programming Guide* . NVIDIA, 2021.

11 Produtos Alcançados

Participação no INTEGRA UFMS 2021 com uma apresentação em vídeo de 5 minutos contendo a explicação do trabalho alcançado.