UFPR – Departamento de Informática

CI1316 – Programação Paralela - Trabalho Prático MPI

Prof. Marco Antonio Zanata Alves

Com base na implementação sequencial em linguagem C do algoritmo **knapsack** listados no final deste arquivo. Leia atentamente e elabore um relatório de no máximo 8 páginas com os seguintes itens:

1. Recorte o kernel (parte principal) de cada algoritmo e explique em suas palavras o funcionamento sequencial do trecho.
2. Explique qual a estratégia final (vitoriosa) de paralelização você utilizou.
3. Descreve a metodologia que você adotou para os experimentos a seguir. Não esqueça de descrever também a versão do SO, kernel, compilador, flags de compilação, modelo de processador, número de execuções, etc.
4. Com base na execução sequencial, meça e apresente a porcentagem de tempo que o algoritmo demora em trechos que você não paralelizou (região puramente sequencial).
5. Aplicando a Lei de Amdahl, crie uma tabela com o speedup máximo teórico para 2, 4, 8 e infinitos processadores. Não esqueça de explicar a metodologia para obter o tempo paralelizável e puramente sequencial.
6. Apresente tabelas de speedup e eficiência. Para isso varie o número de threads entre 1, 2, 4 e 8. Varie também o tamanho das entradas, tentando manter uma proporção. Veja um exemplo de tabela:

		1 CPU	2 CPUs	4 CPUs	8 CPUs	16 CPUs
Eficiência	N=10.000	1	0,81	0,53	0,28	0,16
	N=20.000	1	0,94	0,80	0,59	0,42
	N=40.000	1	0,96	0,89	0,74	0,58

7. Analise os resultados e discuta cada uma das duas tabelas. Você pode comparar os resultados com speedup linear ou a estimativa da Lei de Amdahl para enriquecer a discussão.
8. Seu algoritmo apresentou escalabilidade forte, fraca ou não foi escalável? Apresente argumentos coerentes e sólidos para suportar sua afirmação.
9. Pense sobre cada um dos resultados. Eles estão coerentes? Estão como esperados? A análise dos resultados exige atenção aos detalhes e conhecimento.

Cuidados gerais para efetuar os experimentos

- Para assegurar a corretude da implementação paralela, deve-se verificar se os resultados paralelos batem com os sequenciais executando diferentes entradas. **Lembre-se que o resultado estar igual em uma execução não significa obrigatoriamente que o código está correto**.
- Execute pelo menos 20x cada versão para obter uma média minimamente significativa. Ou seja, todo teste, onde mudamos o número de processos ou tamanho de entrada, devemos executar 20x. Mostrar no relatório a **média com desvio padrão**.
    - As métricas deverão ser calculadas encima da média das execuções.
- Sugiro escolher um modelo de máquina e sempre utilizar o mesmo modelo até o final do trabalho.
    - Cuidar para não executar em servidores virtualizados ou que contenham outros usuários (processos ativos) utilizando a mesma máquina. Diversos servidores do DINF são máquinas virtualizadas e os testes de speedup não serão satisfatórios/realísticos.
    - Cuide para que não haja outros processos ou usuários usando a máquina no mesmo momento que você esteja executando seus testes.
    - Sempre execute com as flags máximas de otimização do compilador, exemplo -O3 para o gcc, afinal queremos o máximo desempenho.
    - Podemos pensar se queremos modificar as configurações de DVFS, também conhecido como turbo-boost, ou seja, fixar a frequência de operação de nossa máquina.
    - Por fim, ainda podemos ter maior controle do experimento, reduzindo a variabilidade ao fixar as threads nos núcleos de processamento.
- **Teste de escalabilidade forte**: Manter um tamanho de entrada N qualquer, e aumentar gradativamente o número de processos. Sugere-se que escolha-se um N tal que o tempo de execução seja maior ou igual a 10 segundos.
- **Teste de escalabilidade fraca:** Aumentar o tamanho da entrada proporcionalmente com o número de processos. Exemplo: 1xN, 2xN, 4xN, 8xN, 16xN. Atenção, escalar N com o número de threads/processos (não de máquinas no caso do MPI).
- Seu **algoritmo deve ser genérico** o suficiente para executar com 1, 2, 3, N threads/processos.
- Ambos os códigos (sequencial e paralelo) **devem gerar as mesmas saídas**.
- Evite figuras e tabelas complexas, opte por formas de apresentação de fácil entendimento.

Regras Gerais de Entrega e Apresentação

A paralelização dos códigos deve ser feita em C ou C++ utilizando as rotinas MPI. A entrega será feita pelo Moodle dividida em duas partes

- **Relatório em PDF (máximo 8 páginas, fonte arial/verdana 12pts.)**
- **Código fonte paralelo (MPI)**

- Casos não tratados no enunciado deverão ser discutidos com o professor.
- Os trabalhos devem ser feitos individualmente.
- **A cópia do trabalho (plágio), acarretará em nota igual a Zero para todos os envolvidos.**
- **Os trabalhos deverão ser apresentados ao vivo no Zoom pelo aluno com vídeo e áudio. A nota irá considerar domínio do tema, robustez da solução e rigorosidade da metodologia.**

Bounded 0-1 Knapsack Problem

Problema de mochila 0-1 limitada

O problema da mochila limitada é um problema clássico de alocação de recursos com muitas aplicações do mundo real, como investimento de capital, síntese de microchips e criptografia, etc. O problema é o seguinte.

Suponha n itens x_1 a x_n onde x_i tem peso w_i e valor v_i . Além disso, suponha uma mochila com capacidade máxima de peso M . Todas as variáveis anteriores são inteiros não negativos.

O problema é determinar valores para todos os x_i , tais que:

$$\sum_{i=1}^{n} x_i v_i \text{ é o maior e}$$

$$\sum_{i=1}^{n} x_i w_i \leq M \text{ é verdade}$$

O problema da mochila limitada é NP-difícil. O problema é dito “limitado” porque não pode se repetir um item x_i .

Esse problema é conhecido como 0-1 pois os itens não podem ser quebrados ao meio, ele pertence ou não a mochila.

Entrada

A entrada contém apenas um caso de teste. A primeira linha contém dois inteiros, separados por um espaço: o número do item n e a capacidade da mochila M . As n linhas restantes, uma por item x_i , também contêm dois inteiros separados por espaço, o valor $(1 \leq v_i < 1024)$ e o peso $(1 \leq w_i < 1024)$ do item x_i .

A entrada deve ser lida a partir da entrada padrão.

Saída

A saída contém apenas uma linha que imprime o inteiro que é o valor máximo alcançado para o problema da mochila.

A saída deve ser gravada na saída padrão.