|  | **Pontifícia Universidade Católica de Minas Gerais****Instituto de Ciências Exatas e Informática****Curso de Ciência da Computação** **Disciplina:** Seminários II - **Prof.** Saulo Augusto de Paula Pinto |
| --- | --- |

**Faça o que é pedido e poste no SGA até 23:59h do dia 03/10/2017.**

Use a mesma máquina em todos os testes pedidos quando for gerar os dados para reportar. Reporte as características principais da mesma: sistema operacional / versão, processador, número de núcleos, tamanho da RAM e *cache*.

Em todos os testes, use o arquivo de dados a ser disponibilizado no SGA

0. Paralelize o código usando o construtor paralelo "for" do OpenMP de duas formas:

1) Versão 0.1: paralelize o for mais externo, apenas

2) Versão 0.2: paralelize os dois for mais internos, apenas

1. Marque o tempo de execução para treinar um perceptron executando mil épocas com uma, duas, quatro, oito,... 128 threads (pseudocódigo e extrato de código seguem) e tabule os dados com seu código.

2. Versão 2: faça o mesmo que no número 1 para o perceptron sequencial (sem threads lançadas pelo OpenMP).

Observe que o objetivo de executar a versão sequencial e a versão "paralela" com uma thread apenas é verificar se a utilização do OpenMP impõe algum *overhead* (sobrecarga) na execução. Use a mesma função para marcar o tempo [omp\_get\_wtime()].

3. Tabule seus resultados para comparar as versões 0.1, 0.2 e 2. Houve diferença significativa? Comente.

/\*

Esta "constante simbólica" serve para dar confiabilidade à medição do tempo, já que os sistemas operacionais têm concorrência entre os processos e o tempo de execução variam de rodada para rodada.

\*/

#define REPETICOES 100

para numThreads de 1 até 128 faça

omp\_set\_num\_threads(numThreads); // seta o número de threads

para v de 1 até REPETICOES faça

tempoInicial = omp\_get\_wtime();

// código do treinamento do perceptron entra aqui com critério de parada igual

// a mil épocas, ou seja, mil iterações sobre a matriz de entrada. Lembre-se de

// reinicializar os pesos para os mesmos valores iniciais sempre a cada rodada!

tempoFinal = omp\_get\_wtime();

somaTempo = somaTempo + (tempoFinal - tempoInicial);

fim para

tempo = somaTempo / REPETICOES; // média do tempo, em segundos

fim para threads

Tabela:

Número |

de |

Threads | 1 2 4 8 16 32 64 128

------------------------------------------------------------------------

0.1 | | | | | | | | |

------------------------------------------------------------------------

0.2 | | | | | | | | |

------------------------------------------------------------------------

Sequencial | |