CI316 – Programação Paralela - Trabalho Prático
Prof. Marco Antonio Zanata Alves – UFPR – Departamento de Informática

Os trabalhos podem ser feitos:
Grupos de 1 pessoa para alunos de Doutorado
Grupos de 1~2 pessoas para alunos de Mestrado
Grupos de 2~3 pessoas para alunos de Graduação (BCC e/ou IBM).

Parte 1 – Implementação com análise sequencial e modelagem analítica.

Escolha um dos seguintes algoritmos clássicos abaixo, ou discuta com o professor um outro algoritmo que seja interessante.

- **Database cracking (lazy quicksort)**: Database Cracking is an incremental partial indexing and/or sorting of the data. It directly exploits the columnar nature of DBMS. Cracking is a technique that shifts the cost of index maintenance from updates to query processing. The query pipeline optimizers are used to massage the query plans to crack and to propagate this information. The technique allows for improved access times and self-organized behavior.
- **Database join:** combines columns from one or more tables in a relational database. It creates a set that can be saved as a table or used as it is. A JOIN is a means for combining columns from one (self-table) or more tables by using values common to each.
- **Knapsack problem with affinities**: is a problem in combinatorial optimization. Given a set of items, each with a weight and a value, determine the number of each item to include in a collection so that the total weight is less than or equal to a given limit and the total value is as large as possible. It derives its name from the problem faced by someone who is constrained by a fixed-size knapsack and must fill it with the most valuable items.
- **The Traveling-Salesman Problem**: asks the following question: "Given a list of cities and the distances between each pair of cities, what is the shortest possible route that visits each city exactly once and returns to the origin city?" It is an NP-hard problem in combinatorial optimization, important in operations research and theoretical computer science.
- **Minimum Weight Triangulation:** is the problem of finding a triangulation of minimal total edge length.[1] That is, an input polygon or the convex hull of an input point set must be subdivided into triangles that meet edge-to-edge and vertex-to-vertex, in such a way as to minimize the sum of the perimeters of the triangles. The problem is NP-hard for point set inputs, but may be approximated to any desired degree of accuracy. For polygon inputs, it may be solved exactly in polynomial time. The minimum weight triangulation has also sometimes been called the optimal triangulation.
- **Maximal Independent Set:** In graph theory, a maximal independent set (MIS) or maximal stable set is an independent set that is not a subset of any other independent set. In other words, there is no vertex outside the independent set that may join it because it is maximal with respect to the independent set property.
- **Lossless Text Compression**: is a class of data compression algorithms that allows the original data to be perfectly reconstructed from the compressed data. By contrast, lossy compression permits reconstruction only of an approximation of the original data, though this usually improves compression rates (and therefore reduces file sizes).
- **Dijkstra:** is an algorithm for finding the shortest paths between nodes in a graph, which may represent, for example, road networks. It was conceived by computer scientist Edsger W. Dijkstra in 1956 and published three years later.
- **N-Body:** is the problem of predicting the individual motions of a group of celestial objects interacting with each other gravitationally.[1] Solving this problem has been motivated by the desire to understand the motions of the Sun, Moon, planets and the visible stars. In the

20th century, understanding the dynamics of globular cluster star systems became an important n-body problem. The n-body problem in general relativity is considerably more difficult to solve.

- **LU Decomposition:** factors a matrix as the product of a lower triangular matrix and an upper triangular matrix. The product sometimes includes a permutation matrix as well. The LU decomposition can be viewed as the matrix form of Gaussian elimination. Computers usually solve square systems of linear equations using the LU decomposition, and it is also a key step when inverting a matrix, or computing the determinant of a matrix. The LU decomposition was introduced by mathematician Banachiewicz in 1938.
- **The K-Means Clustering Problem**: is a method of vector quantization, originally from signal processing, that is popular for cluster analysis in data mining. k-means clustering aims to partition n observations into k clusters in which each observation belongs to the cluster with the nearest mean, serving as a prototype of the cluster. This results in a partitioning of the data space into Voronoi cells.
- **Ray tracing**: is a technique for generating an image by tracing the path of light through pixels in an image plane and simulating the effects of its encounters with virtual objects. The technique is capable of producing a very high degree of visual realism, usually higher than that of typical scanline rendering methods, but at a greater computational cost. This makes ray tracing best suited for applications where the image can be rendered slowly ahead of time, such as in still images and film and television visual effects, and more poorly suited for real-time applications like video games where speed is critical. Ray tracing is capable of simulating a wide variety of optical effects, such as reflection and refraction, scattering, and dispersion phenomena (such as chromatic aberration).
- **Karatsuba Multiplication**: is a fast multiplication algorithm. It was discovered by Anatoly Karatsuba in 1960 and published in 1962. It is faster than the classical algorithm, which requires n2 single-digit products. For example, the Karatsuba algorithm requires 310 = 59,049 single-digit multiplications to multiply two 1024-digit numbers (n = 1024 = 210), whereas the classical algorithm requires (210)2 = 1,048,576. The Karatsuba algorithm was the first multiplication algorithm asymptotically faster than the quadratic "grade school" algorithm. The Toom–Cook algorithm is a faster generalization of Karatsuba's method, and the Schönhage–Strassen algorithm is even faster, for sufficiently large n.
- **Conway Game of Life**: is a cellular automaton devised by the British mathematician John Horton Conway in 1970. The "game" is a zero-player game, meaning that its evolution is determined by its initial state, requiring no further input. One interacts with the Game of Life by creating an initial configuration and observing how it evolves, or, for advanced "players", by creating patterns with particular properties.
- **Game physics**: involves the introduction of the laws of physics into a simulation or game engine, particularly in 3D computer graphics, for the purpose of making the effects appear more real to the observer. Typically, simulation physics is only a close approximation to real physics, and computation is performed using discrete values.
  There are several elements that form components of simulation physics including the physics engine, program code that is used to simulate Newtonian physics within the environment, and collision detection, used to solve the problem of determining when any two or more physical objects in the environment cross each other's path.
- **Floyd-Warshall**: is an algorithm for finding shortest paths in a weighted graph with positive or negative edge weights (but with no negative cycles). A single execution of the algorithm will find the lengths (summed weights) of the shortest paths between all pairs of vertices. Although it does not return details of the paths themselves, it is possible to reconstruct the paths with simple modifications to the algorithm.
- **Quine McCluskey:** also known as method of prime implicants, is a method used for minimization of boolean functions that was developed by Willard V. Quine and extended by Edward J. McCluskey. It is functionally identical to Karnaugh mapping, but the tabular form makes it more efficient for use in computer algorithms, and it also gives a deterministic way to check that the minimal form of a Boolean function has been reached. It is sometimes

referred to as the tabulation method.

- **K-nearest neighbors algorithm**: (k-NN) is a non-parametric method used for classification and regression.[1] In both cases, the input consists of the k closest training examples in the feature space. The output depends on whether k-NN is used for classification or regression: In k-NN classification, the output is a class membership. An object is classified by a majority vote of its neighbors, with the object being assigned to the class most common among its k nearest neighbors (k is a positive integer, typically small). If k = 1, then the object is simply assigned to the class of that single nearest neighbor. In k-NN regression, the output is the property value for the object. This value is the average of the values of its k nearest neighbors. k-NN is a type of instance-based learning, or lazy learning, where the function is only approximated locally and all computation is deferred until classification. The k-NN algorithm is among the simplest of all machine learning algorithms.
- **Smith-Waterman:** performs local sequence alignment; that is, for determining similar regions between two strings of nucleic acid sequences or protein sequences. Instead of looking at the entire sequence, the Smith–Waterman algorithm compares segments of all possible lengths and optimizes the similarity measure.
- **Enumeration Sort**: does not use compare-exchange. The basic idea behind enumeration sort is to determine the rank of each element. The rank of an element ai is the number of elements smaller than ai in the sequence to be sorted. The rank of ai can be used to place it in its correct position in the sorted sequence.
- **N-Queens Problem:** placing n non-attacking queens on an n×n chessboard, for which solutions exist for all natural numbers n with the exception of n=2 and n=3.

Faça a implementação sequencial **otimizada** em linguagem C do algoritmo escolhido.

Crie pelo menos 3 tamanhos de entrada (N, 2*N, 4*N), criando também duas variações por tamanho (ou seja, 6 entradas no total). Os tamanhos distintos irão ajudar nos testes de escalabilidade. As duas variações de entrada por tamanho irão ajudar em testes de algoritmos heurísticos.

A menor entrada deverá demorar pelo menos 8 segundos para atingir a solução. Não crie entradas aleatórias no início do programa, para conseguir testar a mesma entrada com diferentes números de threads e também evitar eventuais comparações entre melhor vs. pior caso.

Evite rodar em servidores do DINF, pois a maioria dessas máquinas estão virtualizadas e os testes de speedup não serão satisfatórios. Tente utilizar as máquinas dos laboratórios para os testes. Escolhendo um modelo de máquina e sempre utilizando o mesmo modelo até o final do trabalho.

Com base na implementação sequencial, faça uma **apresentação** contendo:

- Apresentação detalhada do algoritmo, visão geral, funcionalidades e implementação.
- Profile e análise da sua implementação
- Cálculo da complexidade sequencial
- Projeções de escalabilidade forte para 2, 4, 8 16 e infinitos processadores usando lei de Amdahl.
- Análise de código e debug pode ser feita utilizando os softwares *likwid*, *perf*, *gprof*, *time* ou *Vtune (intel).*

Parte 2 – Paralelização e avaliação de desempenho com causas e consequências.

Utilizando as primitivas Pthreads ou OpenMP, faça a paralelização do código escolhido. Para assegurar a corretude da implementação paralela, deve-se verificar se os resultados paralelos batem com os sequênciais.

Com base na implementação paralela, estenda a **apresentação** com as seguintes informações:

- Estratégia de paralelização, particionamento, pontos de sincronização / concorrência e modelo PRAM.
- Indicação dos pontos de corrida e as dependências reais de dados no código implementado.
- Avaliação de desempenho do código, utilizando as métricas clássicas (overhead, speedup, eficiência, tempo médio, desvio padrão).
- Cálculo do speedup máximo teórico utilizando as leis de Amdahl e Gustafson Barsis.
- Gráficos comparando as medidas obtidas com as estimativas.
- Análise com visão crítica do código e os resultados obtidos. Indique as fontes de ganho e queda de desempenho.