UNIVERSIDADE FEDERAL FLUMINENSE

LEONARDO SANTANA VIEIRA

USO DA BIBLIOTECA CUSP EM CUDA PARA IMPLEMENTAÇÃO DO ALGORITMO DUAL SCALING EM DADOS DE ORDEM DE CLASSIFICAÇÃO

Niterói

2019LEONARDO SANTANA VIEIRA

USO DA BIBLIOTECA CUSP EM CUDA PARA IMPLEMENTAÇÃO DO ALGORITMO DUAL SCALING EM DADOS DE ORDEM DE CLASSIFICAÇÃO

Trabalho de Conclusão de Curso submetido ao Curso de Tecnologia em Sistemas de Computação da Universidade Federal Fluminense como requisito parcial para obtenção do título de Tecnólogo em Sistemas de Computação.

Orientador:

ALTOBELLI DE BRITO MANTUAN

NITERÓI

2019Folha reservada para a ficha catalográfica

LEONARDO SANTANA VIEIRA

USO DA BIBLIOTECA CUSP EM CUDA PARA IMPLEMENTAÇÃO DO ALGORITMO DUAL SCALING EM DADOS DE ORDEM DE CLASSIFICAÇÃO

Trabalho de Conclusão de Curso submetido ao Curso de Tecnologia em Sistemas de Computação da Universidade Federal Fluminense como requisito parcial para obtenção do título de Tecnólogo em Sistemas de Computação.

Niterói, \_\_\_ de \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ de 2019.

Banca Examinadora:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Prof. Altobelli de Brito Mantuan, MSc. – Orientador

UFF - Universidade Federal Fluminense

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Prof. ou Profa. <NOME>, <Título>. – Avaliador

<Sigla da Universidade> - <Nome da Universidade>

Dedico este trabalho à minha mãe.

AGRADECIMENTOS

A meu Orientador Altobelli de Brito Mantuan pelo estímulo e atenção que me concedeu durante o curso.

Aos Colegas de curso pelo incentivo e troca de experiências.

A todos os meus familiares e amigos pelo apoio e colaboração.

“Quem não sabe o que busca, não identifica o que acha”.

Immanuel Kant

RESUMO

Atualmente vivemos um cenário em que é cada vez mais necessário otimizar a forma como tratamos os dados disponíveis, isto devido ao substancial aumento no volume destes para serem analisados e transformados em informação, por isso se faz necessário o desenvolvimento de novas técnicas mais eficientes para tratar grandes volumes de dados. O Dual Scaling é uma dessas técnicas e tem por objetivo processar os itens de uma base de dados e apresentar os resultados de forma simples e precisa. Entretanto, o modelo matemático utilizado por esta técnica é altamente custoso e o fato de só existirem implementações sequenciais disponíveis no mercado somente amplia este problema. Neste trabalho, é utilizada a plataforma de computação paralela Cuda e a biblioteca Cusp, além da linguagem de programação C++, para o desenvolvimento de uma implementação paralela do algoritmo de Dual Scaling. Após o detalhamento da solução, são realizados testes comparando o tempo de execução desta solução a uma implementação sequencial em C++ utilizando a biblioteca Eigen. Por fim, são apresentadas algumas ideias para a continuidade deste estudo.

Palavras-chaves: Dual Scaling, Cuda, Cusp, C++ e programação paralela.

ABSTRACT

We are currently living in a scenario in which it is increasingly necessary to optimize the way we treat the available data, due to the substantial increase in the volume of these data to be analyzed and transformed into information, so it is necessary to develop new techniques more efficient to handle large volumes of data. Dual Scaling is one of these techniques and aims to process the items in a database and present the results simply and accurately. However, the mathematical model used by this technique is highly costly and the fact that there are only sequential implementations available in the market only amplifies this problem. In this work, the Cuda parallel programming platform and the Cusp library, in addition to the C ++ programming language, are used to develop a parallel implementation of the Dual Scaling algorithm. After the solution is detailed, tests are performed comparing the execution time of this solution to a sequential implementation in C ++ using the Eigen library. Finally, some ideas for the continuity of this study are presented.

Key words: Dual Scaling, Cuda, Cusp, C++ e Parallel Programming.

LISTA DE ILUSTRAÇÕES

[Figura 1: Exemplo de Figura. 19](#_Toc378694305)

LISTA DE TABELAS

[Tabela 1 - Ordem de classificação 21](#_Toc19804788)

LISTA DE GRÁFICOS

[Gráfico 1: Exemplo de um gráfico 18](#_Toc378694318)

LISTA DE ABREVIATURAS E SIGLAS

LDB – Lei de Diretrizes e Bases

MALO – Museu ao Ar Livre de Orleans

SMA – Secretaria Municipal de Administração

SME – Secretaria Municipal de Educação

SUMÁRIO

[RESUMO 7](#_Toc378694359)

[ABSTRACT (opcional) 8](#_Toc378694360)

[LISTA DE ILUSTRAÇÕES 9](#_Toc378694361)

[LISTA DE TABELAS 10](#_Toc378694363)

[LISTA DE GRÁFICOS 11](#_Toc378694364)

[LISTA DE ABREVIATURAS E SIGLAS 12](#_Toc378694365)

[1 INTRODUÇÃO 15](#_Toc378694366)

[2 DESENVOLVIMENTO DO TRABALHO 16](#_Toc378694367)

[2.1 TEXTO DO TRABALHO 16](#_Toc378694368)

[2.1.1 REFERÊNCIAS BIBLIOGRÁFICAS 17](#_Toc378694369)

[2.1.2 CITAÇÕES 17](#_Toc378694370)

[2.1.3 IDIOMA ESTRANGEIRO 18](#_Toc378694371)

[2.1.4 FIGURAS, GRÁFICOS E TABELAS. 18](#_Toc378694372)

[2.1.5 NOTAS ENTRE O ORIENTADOR E O ALUNO 19](#_Toc378694373)

[CONCLUSÕES E TRABALHOS FUTUROS 20](#_Toc378694374)

[REFERÊNCIAS BIBLIOGRÁFICAS 21](#_Toc378694375)

[ANEXOS 22](#_Toc378694376)

# INTRODUÇÃO

Os dados hoje em dia são produzidos constantemente e em volumes cada vez maiores, empresas dos mais diversos ramos trabalham intensamente com grandes quantidades de dados gerados a partir de transações de seus negócios. E uma empresa para se manter competitiva precisa buscar formas de analisar esses dados e produzir informações úteis a seu negócio da forma mais eficiente possível.

Para atender a essa necessidade, são necessárias ferramentas que sejam capazes de processar esse volume crescente de dados. Essas ferramentas estão em constante evolução, sempre surgindo novas ou aprimorando as existentes através do desenvolvimento e implementação de novas técnicas, algoritmos e modelos matemáticos, sempre com o objetivo de se processar os dados da forma mais eficiente possível.

Nesse cenário, os modelos matemáticos são de especial importância, devido a capacidade destes de analisar uma base de dados de forma eficiente, extraindo e relacionando os dados utilizando um menor número de transações. Mas ao mesmo tempo, os modelos matemáticos estão se tornando cada vez mais complexos e consequentemente, exigindo cada vez mais poder computacional para a sua execução.

Para solucionar o problema da exigência cada vez maior de poder computacional, uma das propostas existentes é a utilização de computação paralela, em especial, a utilização de plataformas de computação paralela GPGPU, que permitem a utilização da GPU para realização de operações que normalmente seriam realizados na CPU. Como uma GPU possui uma arquitetura altamente paralela, contendo um número muito superior de núcleos se comparada a uma CPU, ela é capaz de realizar os cálculos matriciais propostos pelos modelos matemáticos de forma muito mais eficiente que as CPUs, que realizariam esses mesmos cálculos de forma sequencial ou de forma paralela mas sem a mesma eficiência de uma GPU. Dentre as plataformas disponíveis no mercado, a mais utilizada é a Nvidia Cuda.

Este trabalho apresentará uma implementação paralela utilizando a plataforma Cuda de um modelo matemático chamado de Dual Scaling, e proposto por Nishisato. Esse modelo matemático é capaz de modelar um espaço multidimensional através do mapeamento de colunas e transações de uma base de dados, que será utilizado para gerar uma contextualização semântica dos dados.

Atualmente, o mercado só possui implementações sequenciais baseadas em CPU deste modelo, e como se trata de um modelo matemático altamente custoso, a criação de um algoritmo implementando este modelo de forma paralela utilizado GPU é bastante interessante. Para tal, será utilizada a biblioteca Cusp para facilitar a implementação da solução.

Após o detalhamento de solução, este trabalho irá comparar o tempo de execução do algoritmo paralelo proposto com uma solução baseada em CPU desenvolvida utilizando a biblioteca Eigen.

Este trabalho tem por objetivo apresentar uma implementação paralela do modelo matemático chamado de Dual Scaling, demonstrar a sua eficiência através de testes comparando-a outras implementações e disponibilizá-la para a comunidade acadêmica.

O código fonte da solução está disponível no repositório para consulta. <https://github.com/altobellibm/CEDERJ_2019_LEONARDO_SANTANA_VIEIRA>.

# TRABALHOS RELACIONADOS

Como a quantidade de dados que são produzidos está sempre crescendo, isto torna necessária a evolução proporcional do poder computacional para permitir que o desempenho do processamento dos dados ocorra de maneira satisfatória, como isso não é sempre possível, torna latente a necessidade de otimização das técnicas utilizadas para efetuar o processamento.

Esses problemas de desempenho são visíveis quando a implementação baseada em CPU do Dual Scaling é utilizada para processar uma base de dados grande, isso ocorre devido à complexidade do algoritmo que demanda processamento pesado para alcançar o resultado. Neste caso o desempenho não é o ideal, por isso é necessário o desenvolvimento de novas técnicas ou a evolução das técnicas já existente para que seja possível atender a essa demanda.

Como exemplo de implementação baseada em CPU do Dual Scaling para bases de dados classificatórios, cito a Rankcluster, pacote disponível para a linguagem R disponível no The Comprehensive R Archive Network(CRAN)[1]. A linguagem R é um ambiente livre de desenvolvimento de software voltado a aplicações estatísticas.

Este pacote fornece algumas funções importantes, as mais importantes serão descritas abaixo:

* rankclust(): A função principal, responsável pela análise das bases de dados. Esta função só possui um argumento obrigatório, o data, que por sua vez é uma matriz de n transações de classificações ordenadas. A função retorna um objeto da classe ResultTab.
* convertRank(): Converte bases de dados.
* frequence(): Transforma um conjunto de dados brutos em uma matriz de frequência.
* unfrequence(): O oposto de frequence().

Além do Rankcluster, existem outros pacotes disponíveis para R que atacam o mesmo problema como o pmr e o RMallow, mas o Rankcluster é o mais completo pacote disponível no momento.

## DISCUSSÃO

As soluções citadas até agora, sofrem todas do mesmo problema, elas apresentam problemas de desempenho quando são utilizadas para processar bases de dados muito grandes. Isso ocorre devido ao fato de serem implementações sequenciais baseadas em CPU, portanto todas as operações realizadas são processadas na CPU, uma por uma de forma sequencial.

O objetivo deste trabalho é apresentar uma implementação da técnica de Dual Scaling para bases de dados classificatórios que tire proveito de paralelismo para aumentar o seu desempenho. E a melhor forma de paralelizar o processamento de matrizes é utilizando os recursos disponibilizados por uma GPU já que a sua arquitetura altamente paralela permite ganhos consideráveis de desempenho.

# FUNDAMENTAÇÃO TEÓRICA

A adaptação de métodos estatísticos clássicos para atender as particularidades de pesquisas científicas, como nas ciências sociais e comportamentais, é algo que vem ocorrendo a várias décadas. Pesquisas nessas áreas se caracterizam por muitas vezes não trabalharem com dados numéricos e sim com escalas de medidas incertas.

Esses tipos de dados representam um desafio porque as relações entre os dados não são exatas, dificultando a sua compreensão.

Para a computação, dados são expressões gerais que descrevem atributos ou características de uma entidade. Dados que variam de uma entidade para outra ou que variam a longo do tempo para uma mesma entidade são chamados de variáveis, ou seja, variáveis representam atributos ou características de uma entidade que são passíveis de serem medidas e podem assumir diversos valores como a cor dos olhos, idade, gênero e tipo sanguíneo.

A análise de dados tem por objetivo extrair tantas informações quanto for possível de dados brutos, utilizando para tal técnicas lógicas e estatísticas para avaliar esses dados.

O Dual Scaling é um conjunto de técnicas para análise de dados categóricos. Neste capítulo, será apresentada uma descrição dos tipos de dados que serão analisados neste trabalho e do funcionamento das técnicas que compõem o Dual Scaling, além disso, serão apresentados os cálculos realizados pela técnica.

## BASES DE DADOS CATEGÓRICOS

Variáveis categóricas representam os atributos ou características de uma unidade sendo observada, ou seja, uma variável categórica identifica um atributo ou característica de uma unidade sendo observada. Por exemplo, faixa de renda e grau de escolaridade.

As variáveis categóricas podem ser classificadas como dicotômicas, nominais e ordinais.

* Dicotômicas: Variáveis com apenas duas possibilidades de resposta, por isso também chamadas de binárias. Exemplo: Gênero.
* Nominais: Variáveis que identificam um atributo ou característica sem qualquer propriedade em especial. Exemplo: Tipo sanguíneo.
* Ordinais: Variáveis que identificam um atributo ou característica cuja unidade de observação possui propriedades estruturantes, como por exemplo uma ordenação natural. Exemplo: Faixa de renda.

De acordo com Nishisato[2], criador do método, dados categóricos podem ser divididos entre 2 grupos, dados de incidência e dados de dominância.

* Dados de incidência: No grupo de dados de incidência, o Dual Scaling tem sido utilizado para os seguintes tipos de dados: Tabelas de frequência e dados de múltipla escolha.
* Dados de dominância: No grupo de dados de dominância, o Dual Scaling tem sido utilizado para os seguintes tipos de dados: Dados de ordem de classificação e de comparação pareada.

Este trabalho irá focar no grupo de dados de dominância, mais especificamente, irá utilizar o método para avaliar dados de ordem de classificação.

Dados de ordem de classificação são populares em pesquisas psicológicas e são utilizados para identificar as preferências de uma pessoa. Como exemplo, temos uma pesquisa para identificar as preferências de uma pessoa em relação a cores. Nesta pesquisa o consultado preenche um questionário informando ordenadamente a sua preferência entre as cores pesquisadas. Como pode ser visto na tabela abaixo, as linhas representam as pessoas que responderam o questionário enquanto as colunas representam as cores.

Tabela 1 - Ordem de classificação

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Pessoas | Cores | | | | | |
| Amarelo | Azul | Laranja | Roxo | Verde | Vermelho |
| Pessoa 1 | 1 | 2 | 3 | 4 | 5 | 6 |
| Pessoa 2 | 6 | 5 | 4 | 3 | 2 | 1 |
| Pessoa 3 | 2 | 4 | 6 | 5 | 3 | 1 |
| Pessoa 4 | 1 | 3 | 5 | 6 | 4 | 2 |
| Pessoa 5 | 6 | 4 | 2 | 1 | 3 | 5 |
| Pessoa 6 | 5 | 3 | 1 | 2 | 4 | 6 |

## DUAL SCALING

Dual Scaling é um conjunto de técnicas relacionadas para a análise dos mais diversos tipos de dados categóricos ou quantitativos categorizados, incluindo tabelas de contingência, dados de múltipla escolha, dados de ordem de classificação e dados de comparação pareada.

O Dual Scaling foi criado para ser aplicado em tabelas baseadas no modelo de Guttman, em linhas representam os indivíduos e colunas representam os estímulos. Esses dados são adquiridos através de pesquisas a indivíduos de certos grupos, tendo suas preferências armazenadas e depois mapeadas pelo Dual Scaling.

Através do mapeamento, os indivíduos(representados por linhas) e estímulos(representados por colunas) contidos na tabela sob análise são representados como pontos no espaço-solução resultante. Através da visualização da distribuição dos pontos ao longo dos eixos do espaço-solução, pode-se observar como características comuns dentro de um determinado grupo de indivíduos são percebidos pela proximidade dos pontos no espaço-solução. Ao invés disso, características sem denominador comum ficam distantes no espaço-solução.

Ao analisar as respostas dadas por indivíduos a estímulos, o Dual Scaling realiza além da análise escalar dos estímulos, a análise quanto as diferenças individuais de escala. Assim, o Dual Scaling além de lidar com o problema de calcular o tamanho das características, também lida com um problema mais amplo que é a classificação de dados.

Os dados categóricos são bastante comuns nas ciências sociais e comportamentais. Neste trabalho, iremos nos aprofundar nos dados de ordem de classificação e apresentar uma implementação de uma técnica de Dual Scaling para tratar este tipo de dado.

# IMPLEMENTAÇÃO PARALELA DO DUAL SCALING UTILIZANDO GPU

O Dual Scaling tem o objetivo de analisar e reorganizar os dados de forma a extrair informações destes dados. Mas como os cálculos matemáticos feitos pelo Dual Scaling são bastante complexos, isso faz com que as técnicas exijam um alto poder de processamento para serem utilizadas, o que acarreta em um alto valor a ser investido com equipamentos para a sua melhor utilização.

Neste capítulo será descrita uma implementação do Dual Scaling altamente paralela afim de minimizar os problemas de desempenho da implementação tradicional do Dual Scaling. Alcançar esse objetivo se torna possível através da utilização do plataforma Cuda, que permite que os cálculos matriciais sejam realizados paralelamente na GPU.

Primeiramente será oferecida uma breve descrição da plataforma Cuda e das bibliotecas utilizadas no desenvolvimento da solução. Após essa descrição, será apresentado o código da solução.

## GPU

GPU é o nome dado a um microprocessador especializado em processar gráficos. Devido a sua arquitetura altamente paralela, GPUs são mais capazes de manipular gráficos computadorizados do que as CPUs.

Uma das grandes diferenças entre as GPUs e as CPUs é a arquitetura altamente paralela das GPUs, este fato permite que as GPUs tenham um desempenho muito superior em tarefas paralelizáveis.

## GPGPU

GPGPU é o nome dado ao uso das GPUs em tarefas de natureza não gráfica. Como as APIs gráficas são complexas de serem utilizadas para implementação de algoritmos, se tornou necessário o desenvolvimento de linguagens de alto nível que facilitassem esse processo, permitindo aos desenvolvedores que pudessem abstrair os detalhes de implementação de nível mais baixo das APIs gráficas.

Uma das soluções para este problema foi apresentada pela Nvidia com a introdução do Cuda.

## CUDA

Cuda é uma plataforma de computação paralela desenvolvida pela Nvidia que permite que GPUs compatíveis sejam utilizadas para processamento de propósito geral, ou seja GPGPU.

A plataforma Cuda foi desenvolvida de forma a trabalhar com diversas linguagens de alto nível, como C, C++ e Fortran. As aplicações escritas dessa forma executam as suas partes sequenciais na CPU e através de chamadas específicas, executam as partes paralelizáveis na GPU, poupando a CPU desta carga de trabalho e acelerando a execução da aplicação devido ao nível de paralelização da GPU.

Esse modelo de programação é chamado de computação heterogênea, pois o processamento pode ser realizado por ambos os dispositivos independentemente. No caso do Cuda, a CPU também denominada host é onde é executada a aplicação e é responsável por controlar a gestão de dados, realizar a transferência de memória e configurar a execução da GPU. A GPU também denominada device, é responsável pelo processamento paralelo do algoritmo.

Para a utilização deste modelo, faz-se necessário dividir o problema maior em outros menores, isto requer o ajuste dos algoritmos sequenciais existentes de forma que possam fazer uso do processamento paralelo de forma eficiente.

Para a utilização do Cuda, é preciso uma GPU Nvidia compatível com o Cuda, além da instalação do SDK da plataforma, chamado Cuda Toolkit, que acompanha o driver de vídeo, o compilador NVCC, além de diversas bibliotecas integradas de forma a facilitar o desenvolvimento.

## Thrust

O Thrust é uma biblioteca de algoritmos paralelos semelhante a Biblioteca Padrão de Modelos do C++ (STL). A interface de alto nível do Thrust permite ao programador grandes aumentos de produtividade ao desenvolver aplicações de alta performance. A biblioteca oferece funções que permitem a simples alocação de memória entre a CPU e a GPU, além de oferecer uma série de funções paralelas, permitindo assim a implementação de algoritmos complexos com um código simples e de fácil compreensão e manutenção.

## Cusp

O Cusp, baseada na Thrust, é uma biblioteca de álgebra linear esparsa e computação gráfica semelhante a Biblioteca Padrão de Modelos do C++ (STL). O Cusp fornece uma interface flexível e de alto nível para manipular matrizes esparsas e resolver sistemas lineares esparsos. A biblioteca oferece funções que permitem ainda mais a simplificação do código, produzindo aplicações de alto desempenho com um código simples e de fácil compreensão e manutenção.

# CONCLUSÕES E TRABALHOS FUTUROS

Neste capítulo devem ser colocadas as conclusões que o aluno obteve durante a elaboração do trabalho, bem como o que pretende após sua conclusão (especialização, mestrado, aplicar os conhecimentos em alguma área...).

# REFERÊNCIAS BIBLIOGRÁFICAS

1. *CRAN*. (s.d.). Fonte: The Comprehensive R Archive Network: <https://cran.r-project.org/>.
2. Nishisato, S. (Dezembro de 1993). On quantifying different types of categorical. *Psychometrika, 58*(4), 617-629.
3. ANEXOS

A função das duas definições, Anexo e Apêndice, é semelhante, mas com uma grande diferença entre elas: a autoria. O **ANEXO** de um trabalho acadêmico deve ser aquele texto ou documento que **não foi elaborado por você**, tendo como objetivo servir de legitimação. Já o **APÊNDICE** se configura como texto ou documento **elaborado por você**, tendo como objetivo complementar a sua argumentação.

ANEXO A – TÍTULO DO ANEXO A

ANEXO B – TÍTULO DO ANEXO B