UNIVERSIDADE FEDERAL DE GOIÁS
INSTITUTO DE INFORMÁTICA

ROBERTO DE URZÊDA PAIVA

Algoritmo Paralelo para Processamento de Séries Temporais de Sensoriamento Remoto com Aplicação na Classificação do Uso e Cobertura do Solo

Goiânia
2021

UNIVERSIDADE FEDERAL DE GOIÁS
INSTITUTO DE INFORMÁTICA

TERMO DE CIÊNCIA E DE AUTORIZAÇÃO (TECA) PARA DISPONIBILIZAR VERSÕES ELETRÔNICAS DE TESES

E DISSERTAÇÕES NA BIBLIOTECA DIGITAL DA UFG

Na qualidade de titular dos direitos de autor, autorizo a Universidade Federal de Goiás (UFG) a disponibilizar, gratuitamente, por meio da Biblioteca Digital de Teses e Dissertações (BDTD/UFG), regulamentada pela Resolução CEPEC nº 832/2007, sem ressarcimento dos direitos autorais, de acordo com a Lei 9.610/98, o documento conforme permissões assinaladas abaixo, para fins de leitura, impressão e/ou download, a título de divulgação da produção científica brasileira, a partir desta data.

O conteúdo das Teses e Dissertações disponibilizado na BDTD/UFG é de responsabilidade exclusiva do autor. Ao encaminhar o produto final, o autor(a) e o(a) orientador(a) firmam o compromisso de que o trabalho não contém nenhuma violação de quaisquer direitos autorais ou outro direito de terceiros.

1. Identificação do material bibliográfico

[x] Dissertação [ ] Tese

2. Nome completo do autor

Roberto de Urzêda Paiva

3. Título do trabalho

Algoritmo Paralelo para Processamento de Séries Temporais de Sensoriamento Remoto com Aplicação na Classificação do Uso e Cobertura de Solo

4. Informações de acesso ao documento (este campo deve ser preenchido pelo orientador)

Concorda com a liberação total do documento [X] SIM [ ] NÃO[1]

[1] Neste caso o documento será embargado por até um ano a partir da data de defesa. Após esse período, a possível disponibilização ocorrerá apenas mediante:
a) consulta ao(à) autor(a) e ao(à) orientador(a);
b) novo Termo de Ciência e de Autorização (TECA) assinado e inserido no arquivo da tese ou dissertação.
O documento não será disponibilizado durante o período de embargo.
Casos de embargo:
- Solicitação de registro de patente;
- Submissão de artigo em revista científica;
- Publicação como capítulo de livro;
- Publicação da dissertação/tese em livro.

Obs. Este termo deverá ser assinado no SEI pelo orientador e pelo autor.

Documento assinado eletronicamente por **Wellington Santos Martins**, **Professor do Magistério Superior**, em 22/04/2021, às 17:11, conforme horário oficial de Brasília, com fundamento no art. 6º, § 1º, do Decreto nº 8.539, de 8 de outubro de 2015.

Documento assinado eletronicamente por **ROBERTO DE URZEDA PAIVA**, **Discente**, em 22/04/2021, às 17:31, conforme horário oficial de Brasília, com fundamento no art. 6º, § 1º, do Decreto nº 8.539, de 8 de outubro de 2015.

A autenticidade deste documento pode ser conferida no site https://sei.ufg.br/sei/controlador_externo.php?acao=documento_conferir&id_orgao_acesso_externo=0, informando o código verificador **2019973** e o código CRC **EBEAD7A0**.

ROBERTO DE URZÊDA PAIVA

Algoritmo Paralelo para Processamento de Séries Temporais de Sensoriamento Remoto com Aplicação na Classificação do Uso e Cobertura do Solo

Dissertação apresentada ao Programa de Pós–Graduação do Instituto de Informática da Universidade Federal de Goiás, como requisito parcial para obtenção do título de Mestre em Programa de Pós-Graduação em Ciência da Computação.

Área de concentração: Ciência da Computação.

Orientador: Prof. Dr. Wellington Santos Martins

Co-Orientador: Prof. Dr. Sávio Salvarino Teles de Oliveira

Goiânia
2021

Ficha de identificação da obra elaborada pelo autor, através do
Programa de Geração Automática do Sistema de Bibliotecas da UFG.

Paiva, Roberto
Algoritmo Paralelo para Processamento de Séries Temporais de Sensoriamento Remoto com Aplicação na Classificação do Uso e Cobertura do Solo [manuscrito] / Roberto Paiva. - 2021.
LXI, 61 f.

Orientador: Prof. Dr. Wellington Martins; co-orientador Dr. Savio Oliveira.
Dissertação (Mestrado) - Universidade Federal de Goiás, Instituto de Informática (INF), Programa de Pós-Graduação em Ciência da Computação, Cidade de Goiás, 2021.
Bibliografia.
Inclui siglas, gráfico, tabelas, algoritmos, lista de figuras, lista de tabelas.

1. Meta-características. 2. Rapid-DTW. 3. Uso e Cobertura do Solo. 4. Programação Paralela. 5. Classificação. I. Martins, Wellington, orient. II. Título.

CDU 004

UNIVERSIDADE FEDERAL DE GOIÁS

INSTITUTO DE INFORMÁTICA

ATA DE DEFESA DE DISSERTAÇÃO

Ata nº **05/2021** da sessão de Defesa de Dissertação de **Roberto de Urzêda Paiva**, que confere o título de Mestre em Ciência da Computação, na área de concentração em Ciência da Computação.

Aos trinta e um dias do mês de março de dois mil e vinte e um, a partir das catorze horas, via sistema de webconferência da RNP, realizou-se a sessão pública de Defesa de Dissertação intitulada **"Algoritmo Paralelo para Processamento de Séries Temporais de Sensoriamento Remoto com Aplicação na Classificação do Uso e Cobertura de Solo"**. Os trabalhos foram instalados pelo Orientador, Professor Doutor Wellington Santos Martins (INF/UFG) com a participação dos demais membros da Banca Examinadora: Doutor Sávio Salvarino Teles de Oliveira (goGeo - coorientador), membro titular externo; Professor Doutor Laerte Guimarães Ferreira Júnior (IESA/UFG), membro titular externo; Professor Doutor Gustavo Teodoro Laureano (INF/UFG), membro titular interno. A realização da banca ocorreu por meio de videoconferência, em atendimento à recomendação de suspensão das atividades presenciais na UFG emitida pelo Comitê UFG para o Gerenciamento da Crise COVID-19, bem como à recomendação de isolamento social da Organização Mundial de Saúde e do Ministério da Saúde para enfrentamento da emergência de saúde pública decorrente do novo coronavírus. Durante a arguição os membros da banca não fizeram sugestão de alteração do título do trabalho. A Banca Examinadora reuniu-se em sessão secreta a fim de concluir o julgamento da Dissertação, tendo sido o candidato **aprovado** pelos seus membros. Proclamados os resultados pelo Professor Doutor Wellington Santos Martins, Presidente da Banca Examinadora, foram encerrados os trabalhos e, para constar, lavrou-se a presente ata que é assinada pelos Membros da Banca Examinadora, aos trinta e um dias do mês de março de dois mil e vinte e um.

TÍTULO SUGERIDO PELA BANCA

Documento assinado eletronicamente por **SÁVIO SALVARINO TELES DE OLIVEIRA**, **Usuário Externo**, em 31/03/2021, às 16:31, conforme horário oficial de Brasília, com fundamento no art. 6º, § 1º, do Decreto nº 8.539, de 8 de outubro de 2015.

Documento assinado eletronicamente por **Gustavo Teodoro Laureano**, **Professor do Magistério Superior**, em 31/03/2021, às 16:31, conforme horário oficial de Brasília, com fundamento no art. 6º, § 1º, do Decreto nº 8.539, de 8 de outubro de 2015.

Documento assinado eletronicamente por **Wellington Santos Martins**, **Professor do Magistério Superior**, em 31/03/2021, às 16:31, conforme horário oficial de Brasília, com fundamento no art. 6º, § 1º, do Decreto nº 8.539, de 8 de outubro de 2015.

Documento assinado eletronicamente por **Laerte Guimarães Ferreira Júnior**, **Professor do Magistério Superior**, em 31/03/2021, às 21:34, conforme horário oficial de Brasília, com fundamento no art. 6º, §

1º, do Decreto nº 8.539, de 8 de outubro de 2015.

Documento assinado eletronicamente por **ROBERTO DE URZEDA PAIVA**, **Discente**, em 01/04/2021, às 08:58, conforme horário oficial de Brasília, com fundamento no art. 6º, § 1º, do Decreto nº 8.539, de 8 de outubro de 2015.

A autenticidade deste documento pode ser conferida no site https://sei.ufg.br/sei/controlador_externo.php?acao=documento_conferir&id_orgao_acesso_externo=0, informando o código verificador **1912411** e o código CRC **23B1B689**.

Referência: Processo nº 23070.010269/2021-17 SEI nº 1912411

Roberto de Urzêda Paiva

Possui graduação em Ciência da Computação e pós-graduação *Lato Sensu* em Tecnologias para Gestão de Negócios pela Universidade Federal de Goiás (UFG, 2013, 2015). Tem experiencia em na área de Infraestrutura de TI e Segurança da Informação. Trabalha desde 2011 na Universidade Federal de Goiás, atuando com Gestão de TI e apoio técnico a projetos de pesquisa no Laboratório de Processamento de Imagens e GeoProcessamento (LAPIG). Possui experiência em gestão de servidores Linux e Windows, armazenamento e distribuição de dados satelitários e manutenção de clusteres de processamento.

Dedico este trabalho ao meu filho Ian Pimentel Urzêda.

Agradecimentos

Para realização deste trabalho foi necessário bastante apoio. Meus sinceros agradecimentos:

A minha família, minha esposa Rosiméria Pimentel Damacena e meu filho Ian Pimentel Urzêda, pelo apoio e compreensão durante este período.

Aos meus pais, Regina Celi Urzêda e Geraldo Roberto de Paiva, por todo investimento realizado em minha educação.

Ao meu Orientador, Professor Dr. Wellington Santos Martins, pela disponibilidade apoio e suporte durante a realização de todo o trabalho. Ao meu Co-Orientador, Sávio Salvarino Teles de Oliveira, principalmente pelo incentivo e motivação para realização da pesquisa.

A Universidade Federal de Goiás e ao Laboratório de Processamento de Imagens e Geoprocessamento (LAPIG), responsáveis por toda minha formação superior e profissional. Em especial a toda equipe do LAPIG por me ajudar de diversas maneiras para que fosse possível realização da pesquisa.

"A nossa geração tem que escolher o que ela valoriza mais: lucros de curto prazo ou habitabilidade de longo prazo no nosso lar planetário?"

Carl Sagan,
COSMOS, 1980, ep 4.

Resumo

Paiva, Roberto de Urzêda. **Algoritmo Paralelo para Processamento de Séries Temporais de Sensoriamento Remoto com Aplicação na Classificação do Uso e Cobertura do Solo**. Goiânia, 2021. 65p. Dissertação de Mestrado. Instituto de Informática, Universidade Federal de Goiás.

Nos últimos anos, o volume de dados de observação da Terra aumentou significativamente devido ao grande número de satélites orbitando o planeta. Esses dados são utilizados em abordagens de classificação automatizada, gerando produtos de uso e cobertura do solo para diferentes paisagens ao redor do mundo. O *Dynamic Time Warping* (DTW) é um método clássico utilizado para medir a similaridade entre duas séries temporais. Nesse contexto, algoritmos baseados no DTW se tornaram uma abordagem eficiente para lidar com séries temporais de sensoriamento remoto (STSR). Estes algoritmos podem ser usados para gerar meta-características (i.e., novas características derivadas dos dados originais) para incrementar o desempenho de modelos de classificação. No entanto, algoritmos baseados no DTW possuem complexidade quadrática, fazendo com que o tempo de execução e a utilização de recursos computacionais sejam muito elevados, o que dificulta sua utilização em grandes volumes de dados. Buscando contornar essa limitação, este trabalho apresenta uma solução paralela e totalmente escalável para otimizar a construção de meta-características através de STSR. Além disto, são apresentados resultados da aplicação das meta-características geradas no treinamento e avaliação de modelos de classificação baseados no Random Forest, permitindo a avaliação do impacto da utilização de meta-características na classificação automatizada do uso e cobertura do solo. Nossos resultados mostram que ambas as abordagens levaram à melhoria no tempo de execução e acurácia quando comparadas a estratégias e modelos tradicionais.

Palavras–chave

DTW, TWDTW, Rapid-DTW, Meta-características, Programação Paralela, GPU, Séries Temporais, Sensoriamento Remoto, Classificação, Uso e Cobertura do Solo

Abstract

Paiva, Roberto de Urzêda. . Goiânia, 2021. 65p. MSc. Dissertation. Instituto de Informática, Universidade Federal de Goiás.

The increase in satellite launches into Earth's orbit in recent years has generated a huge amount of remote sensing data. These data are used in automated classification approaches, generating land-use and land-cover products for different landscapes around the world. Dynamic Time Warping (DTW) is a well-known computational method used to measure the similarity between time series, it has been explored in several algorithms for remote sensing time series analysis. These DTW-based algorithms are capable of generating similarity measures between the series and pre-established patterns, these measures can be used as meta-features to increase the performance of classification models. However, DTW-based algorithms require a lot of computational resources and have a high execution time, which makes them difficult to use in large volumes of data. Attempting to avoid this limitation, this article presents a parallel and fully scalable solution to optimize the construction of meta-features through remote sensing time series. In addition, results of the application of the meta-features generated in the training and evaluation of classification models based on Random Forest are presented, allowing the evaluation of the impact of the use of meta-features in the automated classification of land-use and land-cover. The results show that both approaches have led to improvements in execution time and accuracy when compared to traditional strategies and models.

Keywords

DTW, TWDTW, Rapid-DTW, Meta-features, Parallel Programming, GPU, Time Series, Remote sensing, Classification, Land use and Land Cover

Sumário

Lista de Figuras **14**

Lista de Tabelas **15**

Lista de Siglas **16**

1 Introdução **17**
- 1.1 Objetivos 19
- 1.2 Contribuições 19
- 1.3 Publicações 19
- 1.4 Organização da Dissertação 20

2 Conceitos e Trabalhos Relacionados **21**
- 2.1 Séries Temporais de Sensoriamento Remoto 21
- 2.2 Classificação automatizada do uso e cobertura do solo 22
  - 2.2.1 Classificação Baseada em Características 22
  - 2.2.2 Classificação Baseada em Similaridade de Séries Temporais 24
  - 2.2.3 Time-Weighted Dynamic Time Warping (TWDTW) 25
- 2.3 Processamento Paralelo em GPUs 27
  - 2.3.1 Arquitetura e Modelo de Programação CUDA 28
  - 2.3.2 Matriz de Programação Dinâmica em Paralelo 30
  - 2.3.3 Parallel-TWDTW (P-TWDTW) 31

3 Rapid-DTW: Processamento paralelo de STSR **36**
- 3.1 Rapid-DTW 36
- 3.2 Considerações 41

4 Experimentação e avaliação **42**
- 4.1 Ambiente de Experimentação 42
- 4.2 Desempenho do Rapid-DTW 42
  - 4.2.1 Resultados e discussões 44
- 4.3 Efetividade das meta-características na classificação do uso e cobertura do solo com Random Forest 48
  - 4.3.1 Cenários de análise e resultados 49
    - Cenário de comparação A 51
    - Cenário de comparação B 53
    - Cenário de comparação C 54

5 Conclusões **56**
5.1 Contribuições 57
5.2 Limitações 57
5.3 Trabalhos Futuros 57

Referências Bibliográficas **59**

Lista de Figuras

2.1 Série Temporal de Sensoriamento Remoto. Figura adaptada de [40] 22
2.2 Diferenças entre os métodos Distância Euclidiana e DTW [14] 24
2.3 Séries temporais de Soja-Milho e Soja-Milheto com comportamentos semelhantes, porém, cultivadas em diferentes períodos. 25
2.4 Mapeamento do caminho TWDTW [21] 27
2.5 Escalabilidade em CUDA [43] 28
2.6 Hierarquia entre grids, blocos e threads [43]. 29
2.7 Dependência de dados da MPD do método DTW [18] 30
2.8 Técnicas de paralelismo para a computação de matrizes de programação dinâmica [63] 30
2.9 Cada casamento entre Série temporal (St) e Padrão (P) é uma sub-matriz com cruzamento único, e cada sub-matriz é computada através da técnica *Wavefront* em diagonais. 33
2.10 Computação de uma matriz 7x6 com o P-TWDTW 35

3.1 Fluxo de computação da MPD no Rapid-DTW. 37
3.2 Comportamento das *threads* dentro de cada janela. 37
3.3 Computação da MPD 7x6 com o Rapid-DTW 38

4.1 Tempo de execução da MPD com o Rapid-DTW e P-TWDTW. Padrões de tamanho 24 e séries temporais de tamanho 50. 44
4.2 Comparação de tempos de execução completos para geração de meta-características dos algoritmos avaliados. Os números em azul, próximos aos resultados do Rapid-DTW, representam o tamanho da janela utilizada no Rapid-DTW, variando entre 2 e 24. Eixo *y* em escala logarítmica. 45
4.3 Porcentagem de iterações com todas as *threads* trabalhando em paralelo durante a computação da MPD. 46
4.4 Comparação de tempos de execução completos para geração de meta-características do TWDTW em C++ e do Rapid-DTW. Os números em azul representam o tamanho da janela utilizado no Rapid-DTW, variando entre 48 e 384. Eixo *y* em escala logarítmica. 47
4.5 Speedup do Rapid-DTW em relação a versão sequencial TWDTW em C++. 47
4.6 Comportamento das séries temporais das classes analisadas [56]. 49
4.7 Fluxograma da metodologia utilizada para a analisar as meta-características. 51
4.8 Importância das características na classificação 53

Lista de Tabelas

4.1 Tempos de execução e dados estatísticos em relação a execução da computação da MPD, para padrões de tamanho 24 e séries temporais de tamanho 50 44
4.2 Resultados e dados estatísticos do experimento de comparação entre os tempos de execução do Rapid-DTW, P-TWDTW e TWDTW. Os tempos das execuções estão em representados em milissegundos. 46
4.3 Resultados e dados estatísticos da comparação entre o Rapid-DTW e o algoritmo sequencial TWDTW. 48
4.4 Distribuição de amostras entre as classes. 49
4.5 Exemplo de espaço de características 50
4.6 Matriz de confusão do Método Picolli 52
4.7 Matriz de confusão do Método Picolli + meta-características 52
4.8 Comparação entre experimentos Bandas Individuais e Bandas Individuais + meta-características 54
4.9 Matriz de confusão da classificação utilizando apenas as 9 meta-características. 55
4.10 Matriz de confusão da classificação utilizando apenas as 9 meta-características com simplificação das classes de agricultura. 55

Lista de Siglas

CPU	Central Processing Unit
CUDA	Compute Unified Device Architecture
DTW	Dynamic Time Warping
EVI	Enhanced Vegetation Index
GPU	Graphics Processing Unit
KNN	K-Nearest Neighbor
LANDSAT	Land Remote Sensing Satellite
MIR	Mid-infrared
MODIS	Moderate Resolution Imaging Spectroradiometer
MPD	Matriz de Programação Dinâmica
NIR	Near-infrared
NDVI	Normalized Difference Vegetation Index
PD	Programação Dinâmica
P-TWDTW	Parallel-Time Weighted Dynamic Time Warping
SM	Streaming Multiprocessors
STSR	Séries Temporais de Sensoriamento Remoto
TWDTW	Time-Weighted Dynamic Time Warping

CAPÍTULO 1

Introdução

O uso e cobertura do solo apresentam dinâmicas de mudanças que podem ser monitoradas através das análises de Séries Temporais de Sensoriamento Remoto (STSR) [24]. Devido ao grande número de satélites orbitando o planeta, e ao avanço tecnológico dos sensores espacias, as iniciativas de monitoramento estão produzindo grandes volumes de dados de observação da Terra. Estes dados são utilizados para gerar produtos de mapeamento do uso e cobertura do solo, que auxiliam nas decisões relacionadas à segurança alimentar, conservação ambiental, sustentabilidade, emissões de gases de efeito estufa e combate ao desmatamento [42, 7, 2, 20].

As técnicas para mapeamento automatizado do uso e cobertura da solo se tornaram essenciais na era dos grandes volumes de dados de sensoriamento remoto, pois a grande quantidade e diversidade de dados dificulta bastante a utilização de soluções de monitoramento manuais [13]. Diversas abordagens tem projetado modelos de classificação automatizados do uso e cobertura do solo que utilizam algoritmos de aprendizado de máquina e algoritmos de processamento de STSR, alcançando bons índices de acurácia. Melhorar a acurácia destes modelos tem se tornado uma tarefa cada vez mais desafiadora [4, 9, 13]. A análise de similaridade entre STSR tem permitido avanços na construção de bons modelos de classificação do uso e cobertura do solo (i.e., gerar uma meta-característica que representa a distância de uma série temporal para um padrão conhecido de uma classe de uso e cobertura do solo) [39, 40].

O *Dynamic Time Warping* (DTW) é um método clássico, introduzido na década de 1970 [62, 59], capaz de medir similaridade entre séries temporais. Alguns algoritmos baseados no DTW foram desenvolvidos com a finalidade de gerar meta-características baseadas em similaridade entre séries temporais no mapeamento de mudanças do uso e cobertura do solo [28, 61, 54, 31, 3]. Dentre estes algoritmos, o *Time-Weighted Dynamic Time Warping* (TWDTW) [39, 40] se destaca na área de sensoriamento remoto pela criação da função *Time-Weighted*, que torna o método sensível às mudanças climáticas sazonais na vegetação e as dinâmicas de plantio.

Geralmente, os trabalhos que utilizam o TWDTW combinam os resultados de similaridade das STSR com o o algoritmo *k-Nearest Neighbourhood* (*k*-NN) para

realizar a classificação do uso e cobertura do solo [16, 46, 38]. Mas, em algumas regiões geográficas, com alta variabilidade de dados, experimentos comprovaram que a integração entre o TWDTW e o k-NN apresenta baixa acurácia [16]. Alguns trabalhos mostraram ser possível utilizar as meta-características geradas pelos algoritmos baseados no DTW em conjunto com algoritmos de aprendizado de máquina mais sofisticados, reduzido o impacto da variabilidade de dados na classificação [46, 49].

Embora o TWDTW seja um algoritmo baseado no DTW eficaz na geração de meta-características baseadas em similaridade, ele enfrenta problemas que afetam seu desempenho, como o elevado consumo de recursos computacionais e alto tempo de execução, que se agravam com o aumento do tamanho de entrada de dados. Estes problemas ocorrem devido a complexidade quadrática do método DTW, dificultando a utilização desta solução com grandes volumes de dados, o que compromete análises de produtos de STSR de grandes resoluções temporais e espaciais ou até mesmo análises em grandes áreas.

Trabalhos recentes têm explorando o processamento paralelo, buscando formas mais eficientes de se trabalhar com o método DTW para análise de STSR, diminuindo o tempo de execução e tornando-o capaz de trabalhar com volumes de dados maiores [48, 18, 19]. No entanto, estas soluções possuem deficiências de ociosidade de processamento e limitações de escalabilidade. Considerando os avanços tecnológicos para novos sensores espaciais e o surgimento de satélites com altas resoluções temporais e espaciais (p. ex., PlanetScope[1]), estas limitações tendem a prejudicar o uso destas soluções em um futuro próximo.

Nesta dissertação é apresentada uma nova solução paralela para o problema de processamento de grandes volumes de dados de STSR, o algoritmo Rapid-DTW. Este algoritmo é uma solução baseada no método DTW projetada especificamente para trabalhar com dados de sensoriamento remoto, beneficiando-se da função *Time-Weighted*, e sendo capaz de gerar meta-características com alto desempenho. A ideia principal do algoritmo consiste em adaptar técnicas paralelas para a computação da matriz de programação dinâmica com uma estratégia que permita escolher a carga de trabalho realizada por cada unidade de processamento em cada iteração do algoritmo. A proposta foi desenvolvida para o processamento em GPUs, utilizando a arquitetura CUDA, devido a capacidade massiva de paralelismo destes tipos de processadores, e o consumo de energia eficiente com baixos custos de investimento. O Rapid-DTW foi capaz de reduzir significativamente a ociosidade de processamento e eliminar problemas de escalabilidade, quando comparado com as soluções anteriores, obtendo melhores resultados de desempenho.

Este trabalho também apresenta a aplicação das meta-características geradas pelo

[1] https://www.planet.com/

Rapid-DTW, através de um banco de dados de amostras da região do Mato Grosso - Brasil, em modelos de classificação do uso e cobertura do solo baseados no algoritmo Random Forest. As meta-características foram utilizadas em diversos cenários, visando avaliar o impacto de sua utilização na acurácia de modelos que utilizam Random Forest. Os resultados desta experimentação mostraram que em todos os cenários propostos, a utilização das meta-características baseadas em similaridade na composição de modelos de classificação do uso e cobertura do solo foi capaz de gerar melhorias nos resultados de acurácia.

1.1 Objetivos

Os objetivos deste trabalho estão listados a seguir:

- Apresenta uma solução paralela eficiente, que consiga lidar com grandes volumes de dados, para o processamento de STSR no contexto de classificação do uso e cobertura do solo;
- Implementar a solução na linguagem CUDA e avaliar o desempenho, em termos de tempo de execução (*speedup)* e escalabilidade frente a outras soluções;
- Realizar experimentos, com dados reais de sensoriamento remoto, para a geração de meta-características;
- Analisar o impacto de meta-características baseadas em similaridade na classificação do uso e cobertura do solo.

1.2 Contribuições

Neste trabalho são apresentadas as seguintes contribuições:

- O algoritmo paralelo Rapid-DTW, uma solução escalável para geração de meta-características baseadas em similaridade projetado para trabalhar com STSR;
- Implementação em CUDA do Rapid-DTW, disponível em https://github.com/urzedabr/RAPID-DTW;
- Análise do impacto das meta-características geradas pelo Rapid-DTW em relação a modelos de classificação do uso e cobertura do solo baseados em *Random Forest*;
- Avaliação da proposta com dados reais de sensoriamento remoto.

1.3 Publicações

A pesquisa desenvolvida durante a elaboração deste trabalho gerou as seguintes publicações:

- **"P-TWDTW 2.0 – Uma otimização no uso de recursos computacionais para o processamento paralelo de séries temporais de sensoriamento remoto"**. Conferência: Escola Regional de Alto Desempenho do Centro-Oeste (ERAD-CO), 2020, Mato Grosso do Sul, Disponível em: https://sol.sbc.org.br/index.php/eradco/article/view/12646;

- **"Análise de meta-características para classificação de uso e cobertura do solo utilizando Random Forest"**. Conferência: Anais do XI Workshop de Computação Aplicada à Gestão do Meio Ambiente e Recursos Naturais, páginas 71–80, 2020, SBC, DOI: https://doi.org/10.5753/wcama.2020.11021;

- **"*An Efficient Solution to Generate Meta-features for Classification with Remote Sensing Time Series*"**. Conferência: Brazilian Symposium on Geoinformatics (*GeoInfo*), 2020. ISSN 2179-4847. Disponível em: http://urlib.net/rep/8JMKD3MGPDW34P/43PLBGP;

- Através do artigo apresentado no GEOINFO 2020 recebemos um convite para submeter uma versão estendida ao JIDM - *Journal of Information and Data Management*. Foi realizada a submissão do artigo **"*Parallel Processing of Remote Sensing Time Series Applied to Land-Use and Land-Cover Classification*"** no dia 15/02/2021, e encontra-se em processo de revisão.

1.4 Organização da Dissertação

O presente trabalho está organizado da seguinte maneira. O Capítulo 2 apresenta os conceitos fundamentais para a realização da pesquisa, trazendo também os trabalhos que serviram de base para elaboração do trabalho. O Capítulo 3 apresenta o Rapid-DTW, um novo algoritmo para o processamento paralelo de STSR, como foco na geração de meta-características baseadas em similaridade. No Capítulo 4 são detalhados os experimentos realizados com o Rapid-DTW e com as meta-características geradas por ele em modelos de classificação do uso e cobertura do solo. Por fim, o Capítulo 5 apresenta as conclusões, limitações e trabalhos futuros desta pesquisa.

CAPÍTULO 2

Conceitos e Trabalhos Relacionados

Este capítulo apresenta uma abordagem geral sobre a pesquisa literária realizada. Para realização deste trabalho foi necessária a compreensão de conceitos de duas áreas distintas, o Sensoriamento Remoto e a Computação Paralela. Na seção 2.1 é apresentada uma breve introdução as STSR, com foco no produto MOD13Q1. Em seguida, na seção 2.2 são explicadas abordagens que podem ser utilizadas na classificação automatizada do uso e cobertura do solo, apresentando os algoritmos utilizados nestes modelos. Por último, a seção 2.3 apresenta conceitos de processamento paralelo, mostrando o funcionamento da arquitetura CUDA. Nesta seção também é apresentada uma análise crítica do algoritmo P-TWDTW, mostrando os problemas desta solução paralela.

2.1 Séries Temporais de Sensoriamento Remoto

Uma série temporal é uma sequência de observações de determinado fenômeno ao longo do tempo. Essas observações se tornam dados que podem refletir o comportamento, mudanças e características do fenômeno observado [64]. As STSR são obtidas através de sensores espaciais, que são acoplados a satélites lançados na órbita terrestre.

Uma imagem de satélite possui resolução temporal e espacial. A resolução temporal representa a frequência que o satélite visitou e imageou o mesmo local. A resolução espacial define a capacidade de representação de área por pixel de cada imagem. Quanto maior a resolução espacial, maior a qualidade de representação das observações, e quanto maior a qualidade, maior o volume de dados [36]. Uma STSR é composta por um conjunto de imagens obtidas ao longo de um determinado período. A estrutura da STSR é composta por dois eixos espaciais e por um terceiro eixo temporal, formando uma estrutura tridimensional, a Figura 2.1 ilustra a composição de uma STSR.

Existem diversos satélites na órbita terrestre coletando dados de Sensoriamento Remoto diariamente. Vários produtos gerados por esses satélites são utilizados para a classificação de uso e cobertura do solo. Nesta dissertação foi utilizado para realização de análises de classificação do uso e cobertura do solo, dados de STSR do produto

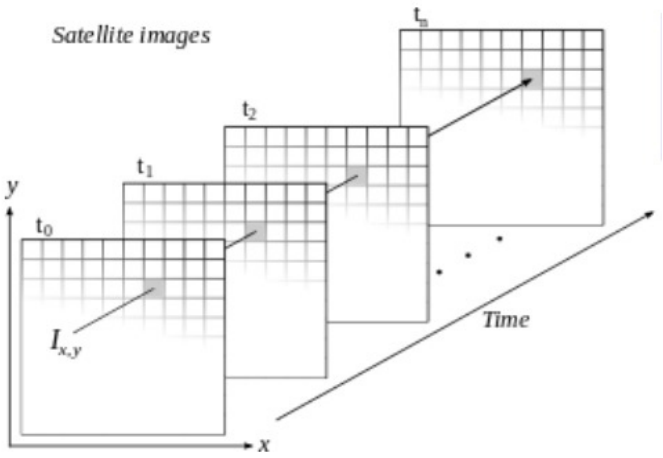


Figura 2.1: Série Temporal de Sensoriamento Remoto. Figura adaptada de [40]

MOD13Q1. Estes dados são obtidos através do sensor MODIS, acoplado ao satélite TERRA.

O sensor MODIS opera com 36 bandas espectrais, abrangendo 2.330 km de largura da faixa de cobertura espacial, está localizado a 705 km de altitude, com diferentes resoluções espaciais (250m, 500m e 1000m). Um bloco de uma imagem MODIS MOD13Q1 consiste em 4800 × 4800 pixels. A frequência de imageamento global ocorre a cada 2 dias com ciclo de repetição de 16 dias. Os dados do produto MOD13Q1 possuem resolução temporal de 16 dias e resolução espacial de 250 metros. O produto MOD13Q1 dispõe dos índices de vegetação, NDVI e EVI, e imagens de reflectância das bandas RED, BLUE, NIR e MIR [32].

2.2 Classificação automatizada do uso e cobertura do solo

As STSR podem ser utilizadas para a classificação de uso e cobertura do solo, assim como gerar características que possam ser utilizadas para identificar determinado tipo de cultura [36]. Diante de diversos métodos para a classificação automatizada do uso e cobertura do solo, podemos identificar duas abordagens que estão se destacando nos trabalhos atuais, a classificação baseada em características e a classificação baseada em similaridade de séries temporais [45].

2.2.1 Classificação Baseada em Características

Diversos métodos de classificação do uso e cobertura do solo em Sensoriamento Remoto fazem uso de algoritmos de aprendizagem de máquina tradicionais. Estes algorit-

mos necessitam de vetores de características para realizar a classificação. Nestes modelos, são extraídas das STSR características para serem utilizadas como dados de entrada para os algoritmos [56, 46].

O uso de algoritmos de aprendizado de máquina para a classificação do uso e cobertura do solo é essencial para a criação de modelos que permitam um mapeamento automatizado e recorrente. Nos últimos anos, os algoritmos *Random Forest* [8] e *Support Vector Machines* (SVM) [66] se tornaram referência de bons classificadores na área de sensoriamento remoto [60, 4, 41].

O *Random Forest* é um dos algoritmos de aprendizado de máquina mais utilizados para classificação de uso e cobertura do solo [50, 4, 27]. Inclusive, é possível observar diversos trabalhos atuais que utilizam o *Random Forest* para o mapeamento de grandes áreas, mostrando sua capacidade em lidar com grandes volumes de dados [51, 65, 52, 1]. A popularidade do *Random Forest* ocorre devido a sua capacidade de atingir acurácia semelhante ao SVM, em conjunto com a facilidade de utilização, com poucos parâmetros a serem configurados pelo usuário e adaptação aos dados de sensoriamento remoto [50].

É necessário um conjunto de características que representem os elementos a serem classificados para o treinamento do *Random Forest*. Este conjunto de características que será a base de dados de entrada para o classificador é chamado de espaço de características [4]. A seleção de características para compor o espaço de características do algoritmo pode ter um impacto crucial no desempenho da classificação [10, 35]. Alguns trabalhos buscam otimizar o espaço de características para classificação de uso e cobertura do solo utilizando métodos de seleção de características [23, 37]. Um espaço de características otimizado pode reduzir significativamente o tempo de processamento e o espaço de armazenamento e, ao mesmo tempo, produzir maior acurácia de classificação do que o conjunto de dados inicial [25].

O trabalho de [56] apresenta uma metodologia de classificação do uso e cobertura do solo que utiliza dados de séries temporais do produto MOD13Q1. No trabalho, os autores propõem a utilização de séries temporais como vetores de entrada para o classificador SVM, neste modelo, os valores das observações anuais de cada banda e índice são utilizados sem nenhum tipo de transformação como composição do espaço de características. Este trabalho obteve melhores resultados de acurácia na classificação de 9 diferentes classes quando comparado a trabalhos anteriores. A abordagem para classificação do uso e cobertura do solo proposta em [56] foi aplicada nesta dissertação utilizando o classificador Random Forest. Além disto, apresentamos também uma extensão do trabalho, adicionando aos espaços de características propostos as meta-características geradas pelo Rapid-DTW.

2.2.2 Classificação Baseada em Similaridade de Séries Temporais

Modelos de classificação baseados na similaridade entre séries temporais utilizam um método matemático para medir a distância entre uma uma série temporal e um padrão pré-estabelecido [5]. No sensoriamento remoto, a partir da medida de similaridade entre as séries, que representam a distância que determinada série se encontra de um padrão, costuma se utilizar algoritmos de classificação baseados em distância nas análises de classificação do uso e cobertura do solo. Diversos trabalhos utilizam o algoritmo k-NN para definição das classes [40, 39, 28, 61, 54, 31].

No sensoriamento remoto, a Distância Euclidiana e o *Dynamic Time Warping* (DTW) são métodos de análises de similaridade entre temporais que são explorados em abordagens para mapeamento a muitos anos [17, 11, 64]. O método da Distância Euclidiana é capaz de identificar a distância entre duas séries temporais de mesmo tamanho, porém, não é capaz de lidar com deslocamento no tempo entre as séries, caso exista. Mais sofisticado, o método DTW é capaz de medir a similaridade entre séries temporais de diferentes tamanhos mesmo que estejam deslocadas no tempo [64].

A Figura 2.2 ilustra o contraste da análise entre os dois métodos. As duas séries temporais possuem bastante semelhanças, porém apenas o método DTW é capaz de perceber essa semelhança, tendo em vista que elas possuem diferentes tamanhos e estão deslocadas no tempo.

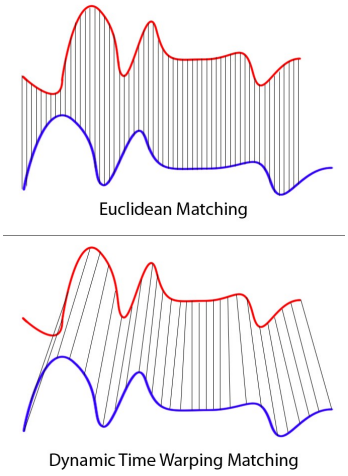


Figura 2.2: Diferenças entre os métodos Distância Euclidiana e DTW [14]

Tanto a Distância Euclidiana, como o DTW não são boas soluções para trabalhar com classificação do uso e cobertura do solo através da anlaise de STSR, tendo em vista que essas séries estão sujeitas a sazonalidade climática e dos diferentes períodos de cultivo de culturas [40]. Por exemplo, uma série temporal de Soja-Milho pode ser muito semelhante a uma série temporal de Soja-Milheto, porém possuem períodos distintos de cultivo (Figura 2.3). Em uma análise com a Distância Euclidiana, o descolamento

de tempo faria com que o método não percebesse a grande semelhança entre as séries, enquanto uma análise com o método DTW trataria as duas séries como muito próximas, fazendo com que fossem classificadas como se fossem apenas uma classe. Na tentativa de solucionar estes problemas um algoritmo específico para medir a similaridade de STSR foi desenvolvido por [40], o *Time-Weighted Dynamic Time Warping* (TWDTW). O TWDTW é um algortimo baseado no DTW, que é capaz de tratar a sazonalidade do cultivo de culturas através de uma função logística baseada no tempo de descolamento (i.e., data inicial e data final) entre as séries temporais comparadas. Na próxima seção o TWDTW será apresentado de forma detalhada.

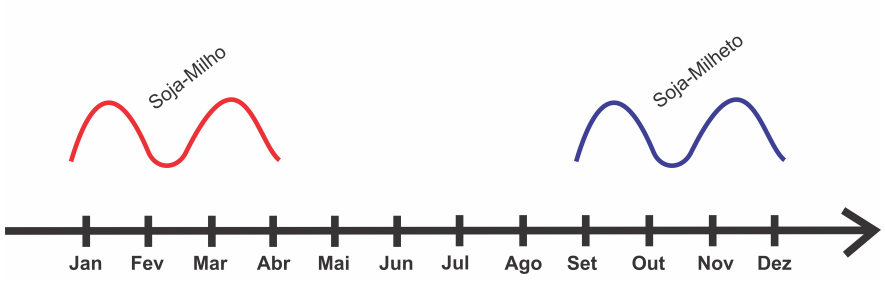


Figura 2.3: Séries temporais de Soja-Milho e Soja-Milheto com comportamentos semelhantes, porém, cultivadas em diferentes períodos.

2.2.3 Time-Weighted Dynamic Time Warping (TWDTW)

Desenvolvido por [39] o algoritmo TWDTW é um algoritmo baseado no método DTW para medir similaridades entre STSR. O TWDTW possui uma função de pesos capaz de lidar com a sazonalidade do cultivo de culturas, denominada *Time-Weighted*. Esta função é responsável por criar uma penalidade para a medida de similaridade de séries temporais e padrões que estejam deslocados no tempo.

Uma série temporal extraída de um determinado tipo de cultura, durante o período inicial do ano, pode ser muito similar a uma série temporal de outro tipo de cultura extraída no final de um ano. Nesse cenário, como há um deslocamento no tempo, o algoritmo penaliza a similaridade entre as duas séries, pois é improvável que sejam o mesmo tipo de cultura devido as dinâmicas de cultivo.

O primeiro passo do algoritmo é realizar o cálculo da matriz de pesos Ψ de dimensão n por m a partir do padrão $U = (u_1, \ldots, u_n)$ e da série temporal $V = (v_1, \ldots, v_m)$. Os elementos $\psi_{i,j}$ da matriz são calculados adicionando um peso ω, gerando a equação $\psi_{i,j} = |u_i - v_j| + \omega_{i,j}$, onde $u_i \in U \; \forall \; i = 1, \ldots, n$ e $v_j \in V \; \forall \; j = 1, \ldots, m$. Para calcular esse peso, que é baseado na distância em dias entre o padrão e a série temporal, é utilizado um modelo logístico com um ponto médio β e uma inclinação α de acordo com a equação (2-1).

$$\omega_{i,j} = \frac{1}{1 + e^{-\alpha(g(t_i,t_j)-\beta)}}, \tag{2-1}$$

onde $g(t_i, t_j)$ é a diferença do tempo em dias entre as datas t_i no padrão U e tj na série temporal V.

A partir do resultado da matriz de pesos Ψ é calculada a matriz de custos acumulados D, através de programação dinâmica, de acordo com a equação (2-2)

$$d_{i,j} = \begin{cases} \Psi_{i,j} + min\{d_{i-1,j}, d_{i-1,j-1}, d_{i,j-1}\} & i \neq 1, j \neq 1 \\ \Psi_{i,j} & i = 1, j = 1 \\ \sum_{k=1}^{i} \Psi_{k,j} & 1 < i \leq n, j = 1 \\ \sum_{k=1}^{j} \Psi i, k & i = 1, 1 < j \leq m \end{cases} \tag{2-2}$$

O k-ésimo caminho de menor custo D gera uma similaridade entre o padrão e uma subsequência de V com a distância associada δ_k, onde a_k é o ponto inicial e b_k é o ponto final da subsequência k. Cada ponto mínimo na última linha da matriz de custo acumulado ($d_{n,j} \; \forall \; j = 1, \ldots, m$), produz um alinhamento, com $b_k = argmin_k(d_{n,j})$, $k = 1, \ldots, K$ e $\delta_k = d_{n,b_k}$, onde K é o número mínimo de pontos na última linha da matriz de custos acumulados D.

A equação (2-3) é utilizada em um algoritmo reverso que mapeia o caminho $P_k = (p_1, \ldots, p_L)$ ao longo do k-ésimo caminho de menor custo em D. O algoritmo inicia em $p_{l=L} = (i = n, j = b_k)$ e termina com $i = 1$ (i.e., $p_{l=1} = (i = 1, j = a_k)$), onde L indica o último ponto do alinhamento. O caminho P_k contém os pontos similares entre as séries temporais. O somatório dos valores destes pontos representa o valor de distância entre as séries temporais comparadas, sendo considerada a medida de similaridade. A Figura 2.4 consiste em uma representação da equação (2-3). Na figura a esquerda (a) podemos observar o destaque do menor caminho, enquanto a direita (b) é apresentada a forma em que a proximidade entre cada ponto do padrão é referenciado na série temporal.

$$p_{l-1} = \begin{cases} (i, a_k = j) & \text{se} \quad i = 1 \\ (i - 1, j) & \text{se} \quad j = 1 \\ argmin(d_{i-1,j}, d_{i-1,j-1}, d_{i,j-1}) & \text{senão} \end{cases} \tag{2-3}$$

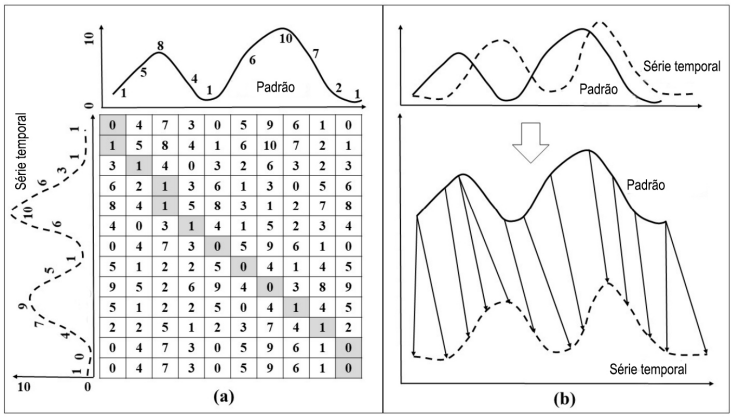


Figura 2.4: Mapeamento do caminho TWDTW [21]

Por padrão, os trabalhos que utilizam o TWDTW combinam os resultados de similaridade das STSR com o algoritmo *k-Nearest Neighbourhood* (k-NN), utilizando $k = 1$ (i.e., o algoritmo classifica as séries temporais de acordo com o maior valor de similaridade entre uma série analisada e os padrões pré-definidos) para realizar a classificação do uso e cobertura do solo [16, 46, 38, 40, 39]. Mas, em algumas regiões geográficas, experimentos comprovaram que a integração entre o TWDTW e o k-NN apresenta baixa acurácia [16]. O trabalho de [46] mostrou ser possível utilizar as meta-características geradas pelos algoritmos baseados no DTW em conjunto com o algoritmo de aprendizado de máquina SVM, obtendo melhorias na acurácia da classificação do uso e cobertura do solo. Esta possibilidade abre caminhos para a utilização das meta-características em diferentes metodologias de classificação do uso e cobertura do solo que utilizem algoritmos de aprendizado de máquina.

2.3 Processamento Paralelo em GPUs

Grandes volumes de dados e algoritmos complexos demandam muitos recursos computacionais para processamento e armazenamento. Um dos maiores desafios da ciência atual é trabalhar com a imensidão de dados disponíveis [34]. A utilização de arquiteturas de processamento paralelo pode se tornar uma abordagem essencial para auxiliar no tratamento destes dados.

Dois modelos de hardware se destacam no processamento paralelo, as arquiteturas *multicore* que correspondem as unidades de processamento (CPU) tradicionais, e as arquiteturas *manycore*, que são as unidades de processamento gráfico (GPU). A GPU (*Graphics Processing Unit*), inicialmente projetada para aceleração de renderização grá-

fica, ganhou espaço nos últimos anos em diversas áreas de pesquisa, produzindo resultados expressivos no processamento em paralelo de grandes volumes de dados [6].

As GPUs atuais se diferem das CPUs por possuir uma capacidade de paralelismo massiva. Essas arquiteturas possibilitam uma enorme vazão de processamento, impactando significativamente no tempo de processamento. Para explorar o poder de processamento desses hardwares, a Nvidia desenvolveu a arquitetura CUDA (*Compute Unified Device Architecture*), que é uma plataforma para desenvolvimento que explora processamento paralelo em GPUs.

O paralelismo pode ser explorado em diferentes tipos de granularidade (razão entre processamento de instruções e comunicação). Algoritmos de granularidade grossa exploram o paralelismo sem se preocupar em paralelizar as tarefas menores da solução. Enquanto algoritmos de granularidade fina possuem maior facilidade no balanceamento de carga, com soluções que executam o paralelismos em tarefas menores do problema [12].

No escopo dessa dissertação é proposto o algoritmo Rapid-DTW, um algoritmo de granularidade fina, que explora o paralelismo de GPUs, visando otimização em termos de performance no processamento de grandes volumes de dados de STSR. O Rapid-DTW foi implementado na linguagem CUDA, utilizando as ferramentas disponibilizadas pela NVIDIA através do framework CUDA Toolkit[1].

2.3.1 Arquitetura e Modelo de Programação CUDA

A arquitetura CUDA consiste em diversos multiprocessadores de fluxo, que são chamados de SMs (Streaming Multiprocessors). Nos SMs são executados blocos de threads em paralelo, e cada bloco é escalonado para qualquer SM disponível, permitindo assim escalabilidade entre aplicações em diferentes GPUs (Figura 2.5).

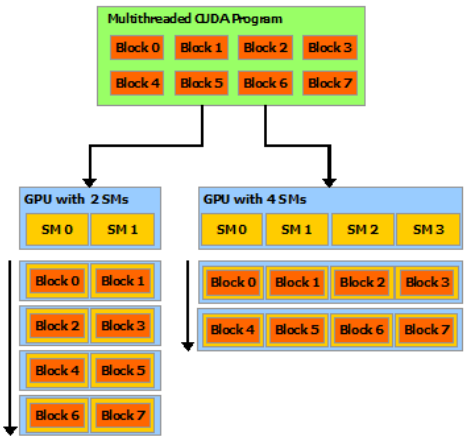


Figura 2.5: Escalabilidade em CUDA [43]

[1] https://developer.nvidia.com/cuda-toolkit

Para trabalhar com os SMs, existem três tipos de estruturas; grids, blocos e threads. Um grid é composto por vários blocos, e cada bloco é composto por um conjunto de threads, essa hierarquia é ilustrada na Figura 2.6 . Uma função que executa na GPU é denominada kernel, onde, ao invocar cada kernel, deve-se enumerar uma quantidade de blocos e threads a serem utilizados por ele. Estas funções, quando chamadas no código, são executadas em paralelo por N threads diferentes.

Os blocos possuem uma memória compartilhada, o que permite que as threads que fazem parte dele cooperem entre si. Para isso é necessário uma sincronização na sua execução para coordenar os acessos à memória. Em CUDA há uma função que faz sincronização por barreira onde todas as threads do bloco irão esperar neste ponto antes de prosseguirem.

O SM cria e executa threads em grupos de 32 threads paralelas chamadas de warps. Com isso, é vantajoso que o número de threads em execução seja múltiplo de 32. Como as threads são agrupados em blocos, elas são executadas em um mesmo SM, compartilhando os recursos disponíveis, o que acaba limitando o número de threads por bloco, que atualmente é de 1024.

O algoritmo Rapid-DTW proposto nesta dissertação busca otimizar a utilização de blocos e threads diante das soluções atuais para processamento paralelo de STSR. A estratégia permite que o algoritmo funcione de forma dinâmica, independente do tamanho de entrada de dados, buscando se adaptar de acordo com a instância do problema, diminuindo a ociosidade de processamento presente em soluções prévias encontradas na literatura.

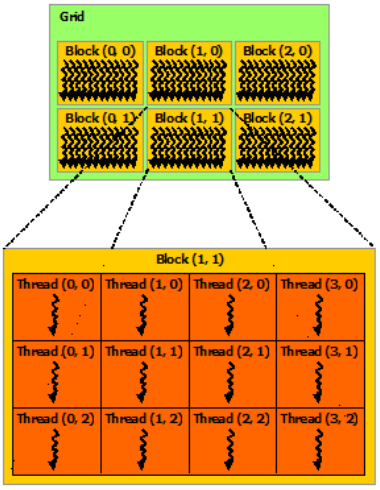


Figura 2.6: Hierarquia entre grids, blocos e threads [43].

2.3.2 Matriz de Programação Dinâmica em Paralelo

A programação dinâmica é uma técnica de resolução de problemas que busca encontrar a melhor solução através de soluções previamente computadas, buscando minimizar o custo computacional para a solução do problema [33]. No cenário de algoritmos baseados no DTW para trabalhar com dados de STSR, os dados escalam de forma espacial e temporal, gerando um enorme volume da dados a serem tratados. A característica temporal destes problemas faz com que haja uma grande dependência de dados, fazendo com que o método DTW dependa do cálculo de uma matriz de programação dinâmica (MPD), a matriz "D". Na computação desta matriz, para que cada cada elemento seja calculado é necessário saber previamente o resultado do cálculo dos elementos anteriores, como pode observado na Figura 2.7.

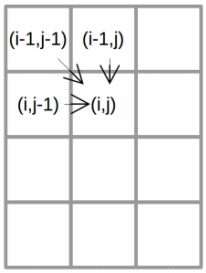
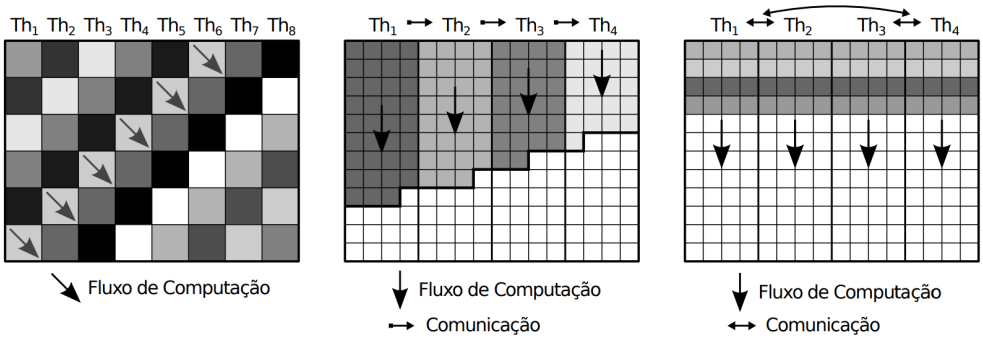


Figura 2.7: Dependência de dados da MPD do método DTW [18]

Existem três técnicas paralelas clássicas para se trabalhar com matrizes de programação dinâmica, *Wavefront* em diagonais, *Wavefront* em blocos e prefixo paralelo [22, 55]. Estas estratégias diferem de acordo com o fluxo de computação e a comunicação entre as threads durante a computação dos elementos da matriz. A Figura 2.8 ilustra as 3 abordagens, mostrando o fluxo computacional e a comunicação entre as threads.



(a) Wavefront (b) Wavefront em Blocos (c) Prefixo Paralelo

Figura 2.8: Técnicas de paralelismo para a computação de matrizes de programação dinâmica [63]

A abordagem mais frequentemente utilizada, por ser mais simples e não depender da comunicação entre as threads, é a estratégia *Wavefront* em diagonais, que realiza a computação das diagonais em paralelo. Nesta estratégia cada thread é responsável por realizar a computação de apenas um elemento diagonal. Os elementos de uma mesma diagonal não possuem dependências entre si, permitindo que todos sejam processados em paralelo. Neste método, o grau máximo de paralelismo (i.e., todas as unidades de processamento trabalhando simultaneamente) ocorre durante a computação da maior diagonal e diminui nos extremos da matriz, gerando uma alta ociosidade de threads, que trabalham todas de forma simultânea apenas na computação das diagonais.

Na técnica de *Wavefront* em blocos a matriz é divida em blocos de elementos, e cada thread fica responsável por computar um destes bloco. A medida que cada thread terminar a computação do seu bloco específico, ela comunica para a thread seguinte que suas dependências foram calculadas, assim iniciando o paralelismo. Esta abordagem diminui o número de theads necessárias para a computação da matriz aumentando a carga de trabalho de cada thread. A carga maior de trabalho faz com que a ociosidade de processamento seja menor em comparação a técnica de *Wavefronto* em blocos, porém, cada bloco da matriz é computado apenas por uma thread [55].

A técnica de prefixo paralelo pode ser utilizada para quebrar dependências de dados, permitindo que todas as threads iniciem simultaneamente a computação de uma mesma linha ou coluna da matriz. A cada iteração, cada thread calcula seus elementos em paralelo utilizando uma equação de recorrência modificada, que permite o cálculo de prefixos máximos locais [63]. No caso de algoritmos baseados no método DTW, a dependência de dados para a computação de um elemento é relacionada ao menor valor entre os elementos previamente calculados (Equação 2-2). Para este tipo de problema, não foi encontrada nenhuma solução de prefixo paralelo na literatura que permita alguma quebra na dependência de dados do problema.

2.3.3 Parallel-TWDTW (P-TWDTW)

Apesar do grande volume de dados gerados pelos satélites atuais e do alto custo computacional dos algoritmos para processamento de STSR, poucos trabalhos utilizam de técnicas de paralelismo visando otimizar o desempenho das análises. Assim como em outras áreas da ciência, abordagens paralelas podem otimizar o tratamento de dados de Sensoriamento Remoto [13].

Dentre os trabalhos que exploram paralelismo no tratamento de dados de Sensoriamento remoto, existe o trabalho de [58], que implementa soluções paralelas para otimização de bancos de dados geográficos. Já o trabalho de [26] busca apresentar estratégias paralelas para o método *Breaks For Additive Season and Trend* (BFAST), utilizado

para detectar quebras em STSR. Para modelos que buscam similaridades entre STSR, apenas os trabalhos [18, 47] apresentam soluções paralelas, através da implementação do algoritmo P-TWDTW.

O algoritmo P-TWDTW é o resultado das primeiras inciativas na exploração de paralelismo em GPUs para otimização de algoritmos baseados no DTW no processamento de STSR [18]. Esta solução possui estratégias paralelas para processar diversas séries temporais simultaneamente e realizar o cálculo da MPD de forma paralela, através da técnica de *Wavefront* em diagonais.

Para realizar a computação de várias meta-características em paralelo, o algoritmo concatena diversos padrões e séries temporais, gerando dois grandes vetores, um de padrões e um de séries temporais. Estes vetores são utilizados para a construção de uma grande matriz, denominada matriz global, que engloba diversas sub-matrizes que representam uma comparação entre um padrão e uma série temporal. Cada par destes irá gerar um valor que representa a distância que aquela série está do padrão. Esta estratégia permite a computação simultânea da medida de similaridade entre diversos pares de padrões e séries temporais, com apenas uma troca de dados da memória da CPU para a GPU, diminuindo significativamente o *overhead* de movimentação de dados entre o *host* e o *device*.

Em seguida cada bloco de *threads* da arquitetura CUDA recebe um par de padrão e série temporal. Dentro de cada bloco cada diagonal da MPD tem seus elementos calculados em paralelo, respeitando a dependência de dados inerente da programação dinâmica. A Figura 2.9 ilustra a estratégia da matriz global, para computação de diversos padrões e séries temporais em paralelo e mostra o funcionamento da técnica de *Wavefront* em diagonais utilizada pelo P-TWDTW. Nesta técnica, o algoritmo computa de forma paralela todos os elementos de uma diagonal, seguindo um fluxo de computação em diagonais.

O P-TWDTW carrega os padrões e séries temporais em filas nas memórias da CPU e GPU. Uma *thread* na CPU fica responsável por organizar as filas, de forma que não haja estouro de memória, otimizando o carregamento e troca de dados entre as memórias da GPU e CPU. Em seguida o algoritmo junta todos os padrões e séries temporais em duas filas, gerando um vetor de séries temporais e um vetor de padrões. A partir destes vetores é computada a matriz de pesos Ψ, de forma trivialmente paralela (i.e., como a matriz de pesos Ψ pode ser computada sem nenhuma dependência de dados, de acordo com a equação (2-1), todos os elementos da matriz podem ser calculados de forma paralela). Estas etapas do estão descritas no Algoritmo 2.1.

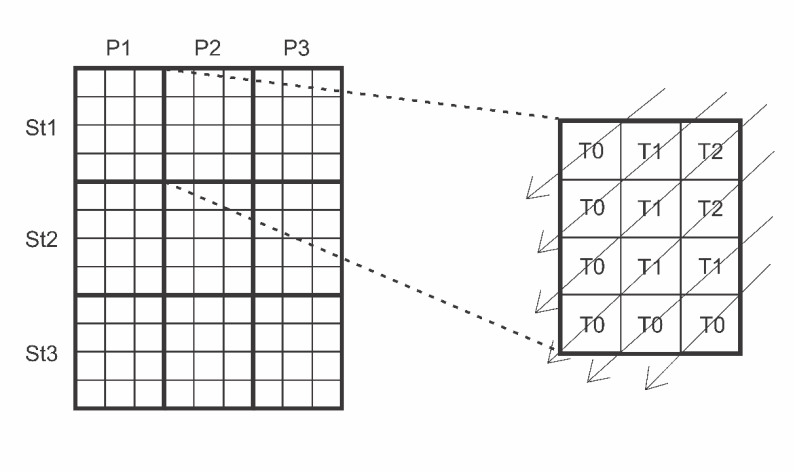


Figura 2.9: Cada casamento entre Série temporal (St) e Padrão (P) é uma sub-matriz com cruzamento único, e cada sub-matriz é computada através da técnica *Wavefront* em diagonais.

Algoritmo 2.1: ptwdtw(*Q*, *bQ*, *max_bQ*, *S*, *bS*, *max_bS*)

Entrada: *Q*: conjunto de padrões *U* a serem utilizados na consulta
bQ: número de padrões *U* processados ao mesmo tempo
max_bQ: número de padrões *U* na memória
S: conjunto de séries temporais *V*
bS: número de séries temporais *V* processadas ao mesmo tempo
max_bS: número de séries temporais *V* na memória
Saída: Matriz de pesos Ψ

```
1  while Tiver mais padrões U e séries temporais V no disco do
2      queueQ ← carrega max_bQ padrões U para a memória RAM
3      queueS ← carrega max_bS séries temporais V para a memória RAM
4      while Tiver mais padrões em queueQ e séries temporais em queueS do
5          gpu_queueQ ← carrega bQ padrões U para a memória global da GPU
6          gpu_queueS ← carrega bS séries temporais V para a memória global da GPU
7          bigQ ← junta todos os padrões U em gpu_queueQ
8          bigS ← junta todos as séries temporais V em gpu_queueS
9          Ψ ← calcula a matriz de custo, de forma trivialmente paralela, entre bigQ e bigS
10     end
11 end
```

A partir da matriz de pesos Ψ, o algoritmo realiza a computação da MPD utilizando a técnica de *Wavefront* em diagonais. Para que esta técnica funcione de forma correta, o número de *threads* alocadas para computar cada sub-matriz (i.e., par de padrão e série temporal) precisa ser igual ao tamanho da diagonal principal, ou seja, o tamanho

dos dados de entrada impacta diretamente no número de threads necessárias para a computação da MPD.

Com estas estratégias, o P-TWDTW consegue realizar a computação de diversas meta-características em paralelo de forma eficiente. Para cada par entre padrão e série temporal é computada uma meta-característica, que representa a distância da série para o padrão, e a matriz global permite a computação de diversos pares em paralelo. A ideia de utilizar uma matriz global permite a computação de grandes quantidades de padrões de séries temporais, controlando a utilização de memória da CPU e GPU. Porém, em relação ao tamanho das séries temporais (i.e., quantidade de observações), que impacta diretamente na computação da MPD, a solução de *Wavefront* em diagonais apresenta alguns problemas.

A estratégia do de *Wavefront* em diagonais para a computação da MPD limita a carga de trabalho de cada *thread* a um elemento da diagonal por iteração, fazendo com que a carga de trabalho de cada *thread* seja muito pequena. As consequências desta baixa carga de trabalho são uma alta ociosidade de processamento e limitações de escalabilidade. A Figura 2.10 exemplifica a ociosidade de *threads* durante a execução da MPD com o P-TWDTW, em uma matriz de 7 linhas e 6 colunas. Para computar esta matriz são necessárias 6 *threads*. Observa-se que todas as *threads* trabalham simultaneamente em paralelo somente em 2 passos (Figuras (f) e (g)), enquanto o algoritmo necessita de 12 iterações para completar a computação da matriz.

Devido a dependência de dados da programação dinâmica, em cada iteração é necessária uma chamada de sincronização, para garantir que as *threads* só processem após terem todas as dependências computadas. Com a baixa carga de trabalho por *thread*, a estratégia em diagonais necessita de uma alta quantidade de iterações para computar a matriz.

O problema de escalabilidade do P-TWDTW também merece atenção. Na estratégia de *Wavefront* em diagonais, a cada iteração, uma diagonal completa é computada, tendo uma thread responsável pela computação de um elemento específico da diagonal. Esta característica faz com que o número de threads necessárias para a computação de cada par precise ser igual a tamanho da maior diagonal, limitando o algoritmo de acordo com a quantidade de threads disponíveis na arquitetura. A arquitetura CUDA/GPU, para a qual o P-TWDTW foi projetado, possui atualmente um limite de 1024 *threads* para cada bloco [44]. Neste caso, o algoritmo não é capaz de computar medidas de similaridade entre padrões e séries temporais quando ambos forem maiores do que 1024.

Esta dissertação apresenta uma solução que aproveita a ideia de computar diversas meta-características em paralelo, utilizando a estratégia de matriz global. Porém, propõe uma nova solução paralela para a computação da MPD, responsável pela escalabilidade temporal do problema. Uma boa solução para a computação da MPD pode

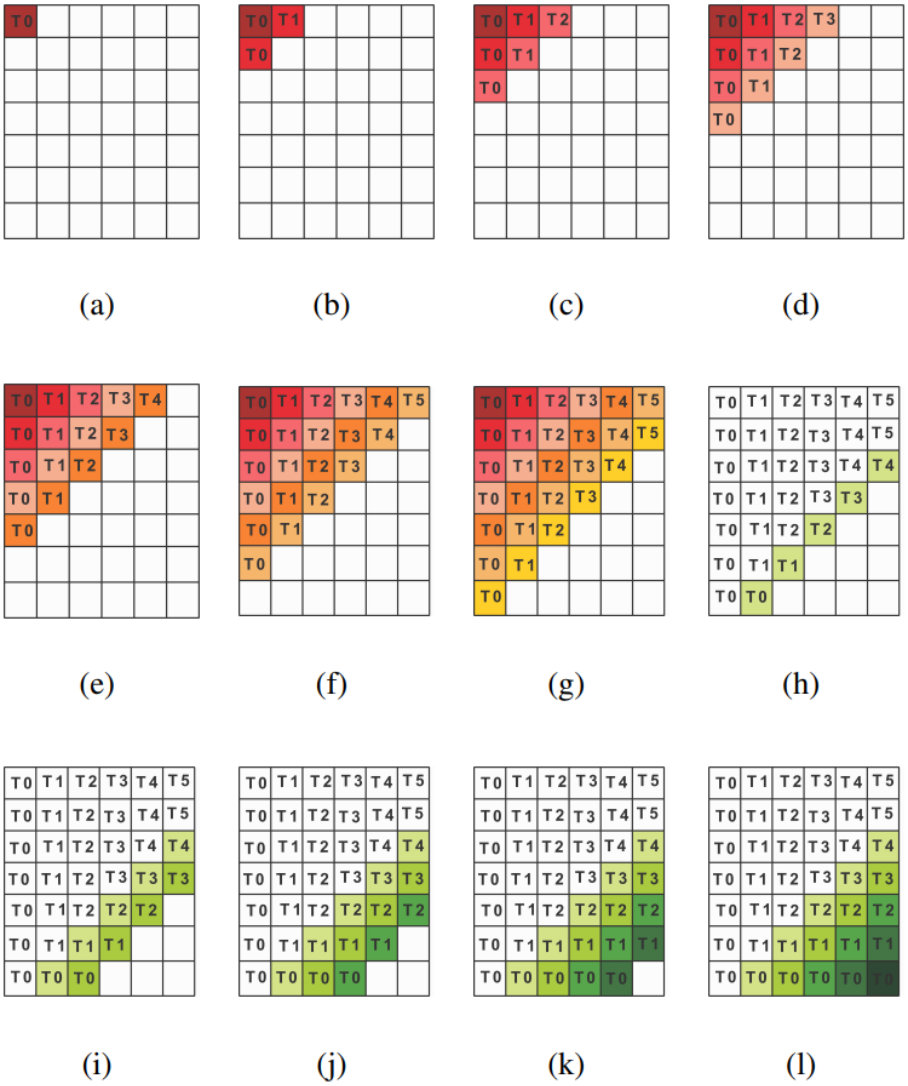


Figura 2.10: Computação de uma matriz 7x6 com o P-TWDTW

melhorar bastante a performance do algoritmo, tendo em vista que algoritmos baseados no DTW possuem complexidade quadrática.

Na seção 3.1 é apresenta a descrição da proposta de um novo algoritmo paralelo de granularidade fina para a geração de meta-características baseadas em similaridade de STSR. O Rapid-DTW é uma solução dinâmica (i.e., pode ser adaptado de acordo com os tamanhos de séries temporais e padrões), que diminui a ociosidade de processamento, tornando possível a atribuição de maneira dinâmica da carga de trabalho das unidades de processamento.

CAPÍTULO 3

Rapid-DTW: Processamento paralelo de STSR

Este capítulo descreve o algoritmo Rapid-DTW, uma nova solução paralela dinâmica para o processamento de STSR, que demanda menor tempo de execução e otimiza o uso de recursos computacionais em relação a soluções prévias do estado da arte. O Rapid-DTW foi projetado para trabalhar especificamente com dados de sensoriamento remoto, aplicando a técnica *Time-Weighted* e explorando paralelismo na arquitetura CUDA/GPU.

3.1 Rapid-DTW

O Rapid-DTW é um novo algoritmo baseado no método DTW. A ideia principal do algoritmo consiste em adaptar as técnicas de *Wavefront* em blocos e *Wavefront* em diagonais para criar uma estratégia em que seja possível escolher a carga de trabalho realizada por cada unidade de processamento em cada iteração do algoritmo. A nova estratégia criada para o a computação da MPD consiste em realizar a computação dos elementos em janelas, de forma que diversas janelas possam ser computadas em paralelo durante uma iteração do algoritmo. Esta computação em janelas permite que o algoritmo experimente diferentes tamanhos de janelas, podendo obter configurações ideais para instâncias específicas do problema. Assim, o Rapid-DTW é capaz de diminuir a ociosidade de threads, o que melhora o uso de recursos da GPU, possibilitando melhor performance em tempo de execução.

Projetado para trabalhar especificamente com STSR, o Rapid-DTW se divide em 4 etapas principais. Primeiramente o algoritmo concatena padrões e séries temporais em filas de forma coordenada, evitando ultrapassar os tamanhos de memórias da CPU e GPU. Esta solução é a mesma utilizada pelo P-TWDTW, com pequenas alterações de implementação (parametrização do código de acordo com a arquitetura, utilização de variáveis que consomem menos memória e eliminação de condicionais não necessários). Em seguida é computada a matriz global de pesos Ψ de forma trivialmente paralela. Nesta etapas temos os pesos computados de acordo com a técnica *Time-Weighted* desenvolvida por [40, 39]. A partir da matriz de pesos Ψ, a estratégia proposta nesta dissertação entra

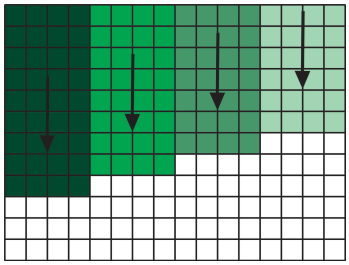


Figura 3.1: Fluxo de computação da MPD no Rapid-DTW.

em execução, realizando a computação da matriz de programação dinâmica "D"de acordo com o tamanho de janela definido para uma entrada de dados específica. Por fim é calculado o caminho de menor custo para cada par de padrão e série temporal.

O principal foco da computação em janelas é otimizar a computação da MPD, etapa que possui o maior custo computacional e tempo de execução em algoritmos baseados no método DTW. Utilizando a abordagem do cálculo dos elementos da MPD em janelas, o fluxo de computação da matriz ocorre de forma semelhante a técnica *Wavefront* em blocos, mas sem a necessidade de dividir a matriz em blocos de elementos. Isso é possível pois a estratégia em janelas é uma técnica hibrida, que combina as soluções de *Wavefront* em diagonais e *Wavefront* em blocos. A Figura 3.1 ilustra o fluxo de computação da MPD no Rapid-DTW.

A estratégia de computação baseada em janelas do Rapid-DTW respeita a dependência de dados inerente da programação dinâmica. Conforme ilustrado na Figura 3.2, dentro de cada janela a unidade de processamento executa seu trabalho de forma sequencial, garantindo a dependência dos dados, enquanto janelas com a mesma cor podem ser processadas em paralelo. A capacidade de escolher o tamanho da janela permite diversificação da carga de processamento, visando otimização do uso de recursos, de acordo com o tamanho da entrada de dados. Quanto maior a janela definida, maior é a carga de trabalho de cada unidade de processamento, o que implica em menor ociosidade de processamento. A capacidade de definir o tamanho da janela a partir do número de threads disponíveis faz com que o Rapid-DTW seja capaz de computar medidas de similaridade entre padrões e séries temporais de tamanhos maiores, melhorando bastante a escalabilidade da solução.

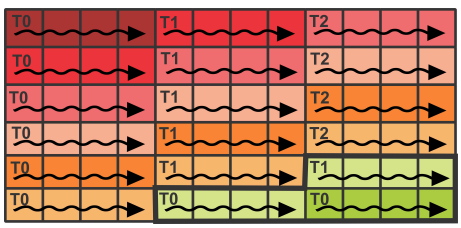


Figura 3.2: Comportamento das *threads* dentro de cada janela.

A solução da computação dos elementos em janelas reduz a ociosidade de threads e o número de iterações do algoritmo em relação a técnica de *Wavefront* em diagonais, pois a estratégia em janelas permite trabalhar com menos threads por bloco, aumentando a carga de trabalho individual de cada uma. Podemos observar na Figura 3.3 a computação da mesma matriz $7x6$ apresentada na seção 2.3.3. Na figura, janelas de elementos de mesma cor são computadas paralelamente, e o tamanho da janela é igual a 2. Observa-se que toda a matriz é computada em apenas 9 passos, e em 5 destes (Figuras (c) a (g)) todas as *threads* trabalham de forma paralela.

O algoritmo da estratégia em janelas para a computação da MPD com o Rapid-DTW está descrito no Algoritmo 3.1. Entre a segunda e a quarta linha são definidas variáveis constantes, isto é realizado com o objetivo de evitar que as operações matemáticas em laços ocorram diversas vezes, otimizando a execução do algoritmo. Na linha 2 é cal-

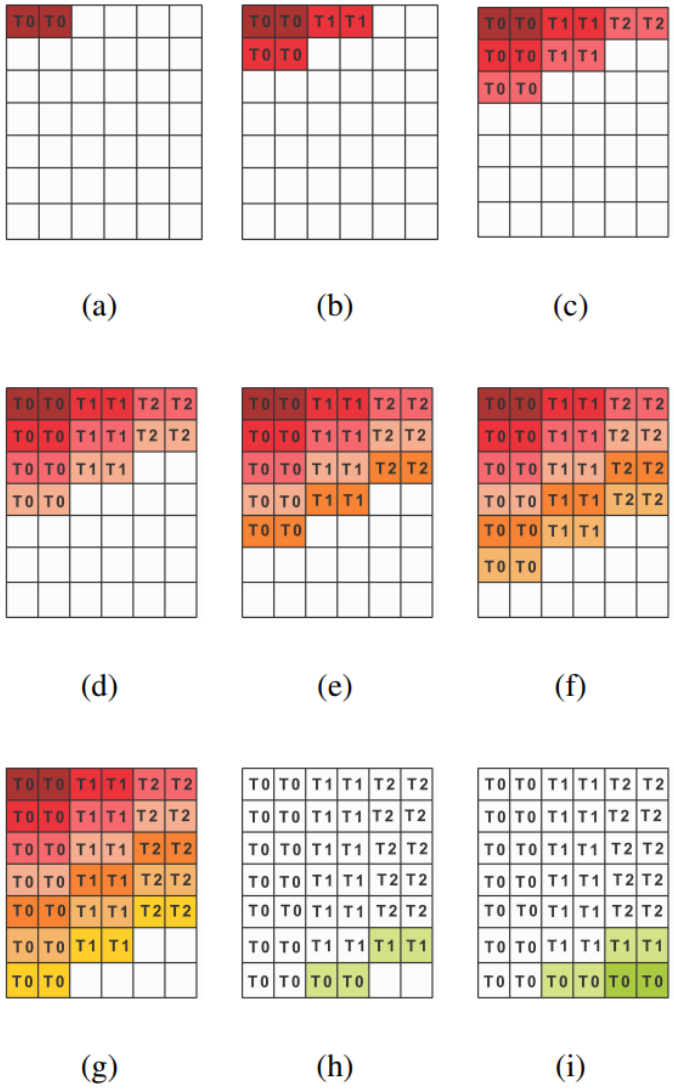


Figura 3.3: Computação da MPD 7x6 com o Rapid-DTW

culado o tamanho da janela de acordo com a quantidade de *threads*. As variáveis *aux* e *auxj* auxiliam na localização dos elementos de cada janela, enquanto a variável *base* é responsável por manter cada *thread* dentro do tamanho de janela especificado.

O laço entre as linhas 5 e 18 realiza a computação da parte superior da matriz, fazendo a indexação de acordo com a quantidade de linhas. Dentro deste laço, nas linhas 6 e 7 temos as variáveis *base* e *auxj*. É necessário garantir que cada *thread* trabalhe, em cada iteração, apenas dentro da janela definida. A variável *base* é responsável por manter a *thread* dentro do tamanho de janela especificado. Enquanto isso, dentro de cada iteração, a variável *j* pode assumir diversos valores de acordo com o tamanho da janela. Em função disto, utilizamos a variável *auxj*, que irá atualizar o o valor de *j* de acordo com o elemento a ser computado.

A linha 8 define uma condição que controla a quantidade de *threads* trabalhando simultaneamente em cada iteração, pois é necessário garantir que cada *thread* só inicie sua computação após as *threads* anteriores finalizarem a computação de sua janela. No final da condição, na linha 17, é realizada a sincronização das *threads*. Na linha 10 temos o laço para realização do cálculos dos elementos da matriz, utilizando a variável *base* para definir onde é inciada a janela de acordo com o *id* da *thread*. Neste laço, são definidos os elementos *i* e *j* a serem atualizados pela função *update_element*(i,j), que atualiza cada elemento de acordo com a equação (2-2) apresentada na seção 2.2.3.

Entre as linhas 21 e 35 é realizada a computação da parte inferior da MPD, de forma análoga a parte superior. Nas linhas 19 e 20, as variáveis *si* e *auxj* são reiniciadas. A variável *si* é necessária para que a *base* possa controlar o tamanho da janela. A variável *aux* irá auxiliar na definição do índice inicial de computação de cada *thread*, tendo em vista que durante a computação dos elementos inferiores da diagonal o valor inicial da variável *j* é diferente para cada *thread*. Diferentemente da computação da parte superior, onde a variável *j* poderia assumir uma quantidade fixa de valores diferentes a cada interação igual ao tamanho da janela, na parte inferior, a quantidade de valores diferentes se torna a subtração do número de colunas pelo tamanho da janela.

A linha 21 tem a mesma função da linha 5 apresentada anteriormente, porém, agora o controle é feito baseado no número de colunas. As linhas 22, 23 e 24 inicializam as variáveis *base*, *aux* e *auxj* que irão a cada interação controlar a quantidade de trabalho de cada *thread* de acordo com o tamanho da janela, e os valores (i,j) a serem computados em cada iteração. Entre as linhas 25 e 33, o algoritmo funciona exatamente como na computação da parte superior apresentada anteriormente, apenas com alterações nas equações de indexação dos elementos.

Algoritmo 3.1: Computação da MPD com o Rapid-DTW

Entrada: Ψ: Matriz de pesos
y: número de linhas da matriz
x: número de colunas da matriz
num_threads: número de *threads*
Saída: Matriz D

```
1  tid ← thread id
2  windowSize ← x/num_threads
3  tidWindow ← tid * windowSize
4  tidWindowaux ← tid * (windowSize − 1)
5  for (si = 0; si < y; si++) do
6      base ← tidWindow + (si − tid) * x
7      auxj ← tidWindowaux
8      if (tid ≤ min(si, x − 1)) then
9          All threads in paralalel do:
10         for (index = base; index < base + windowSize; index++) do
11             i ← si − tid
12             j ← tid + auxj
13             update_element(i, j)
14             auxj ← auxj + 1
15         end
16     end
17     sync_barrier
18 end
19 si ← (y − 1 − tid) * x
20 auxj ← 0
21 for sj ← ((x/windowSize) − 2; sj ≥ 0; sj − −) do
22     base ← tidWindow + si + windowSize + auxj
23     aux ← 0
24     auxj ← auxj + windowSize
25     if (tid ≤ min(sj, y − 1)) then
26         All threads in paralalel do:
27         for (index = base; index < base + windowSize; index++) do
28             i ← y − tid − 1
29             j ← x − (windowSize * sj) − windowSize + aux + tidWindow
30             update_element(i, j)
31             aux ← aux + 1
32         end
33     end
34     sync_barrier
35 end
```

3.2 Considerações

Sabemos que os dados relativos a STSR escalam em duas vertentes, espacial e temporal. A resolução espacial diz respeito ao número de *pixels* que a imagem de satélite possui, representando uma unidade de área. Com o avanço tecnológico dos sensores espaciais, as resoluções espaciais tende a melhorar, possibilitando com que a área de representação de cada pixel seja cada vez menor (p.ex., Os dados MOD13Q1 possuem uma resolução de $250x250$ metros, enquanto os dados *PlanetScope* chegam a apresentar resoluções espaciais de $5x5$ metros). Em consequência, há um aumento da quantidade e *pixels* por imagem, e como cada *pixel* representa uma série temporal, essa vertente é responsável pela escalabilidade em relação a quantidade de STSR.

Já a resolução temporal representa a quantidade de observações realizadas pelo satélite em cada pixel em um determinado espaço de tempo. Sempre que um satélite capta uma imagem de uma região, uma observação é contabilizada. Existem diversos dados de sensoriamento remoto com as mais variadas resoluções temporais, em intervalos que podem variar de minutos a dias. No âmbito dos algoritmo DTW-based, a resolução temporal impacta diretamente na complexidade quadrática do algoritmo (i.e., quanto maior o número de observações, mais recursos computacionais e tempo de execução são necessários).

Uma solução paralela eficiente para geração de meta-características baseadas em similaridade de STSR deve ser capaz de lidar de forma satisfatória com o aumento das resoluções espaciais e temporais simultaneamente. O Rapid-DTW, apresentado neste capítulo, é um algoritmo paralelo de granularidade fina que apresenta um conjunto de ideias capaz de lidar com ambas as vertentes de forma eficiente. Ele aproveita a ideia de uma matriz global para computação de diversas meta-características em paralelo, e otimiza a execução da MPD com a estratégia de computação dos elementos em janelas. Desta forma, o Rapid-DTW se torna uma solução competitiva para a geração de meta-características baseadas em similaridade de STSR explorando paralelismo.

Além do desenvolvimento do Rapid-DTW, esta pesquisa avalia o impacto da utilização das meta-características geradas por ele no espaço de características de modelos de classificação do uso e cobertura do solo. No próximo capitulo, juntamente com a análise de performance do do Rapid-DTW, é apresentada uma metodologia para avaliar esta aplicação em modelos de classificação do uso e cobertura do solo baseados no classificador *Random Forest*.

CAPÍTULO 4

Experimentação e avaliação

Este capítulo apresenta os experimentos realizados com a finalidade de avaliar o desempenho do Rapid-DTW na geração de meta-características, em termos de tempo de execução e capacidade de trabalhar com diversos tamanhos de STSR, e a efetividade destas meta-características na composição de modelos de classificação do uso e cobertura do solo baseados em Random Forest. A seção 4.1 apresenta as especificações da máquina utilizada em todos os experimentos. Nas seções 4.2 e 4.3 são discutidos os experimentos em relação ao desempenho do Rapid-DTW e a efetividade das meta-características, respectivamente.

4.1 Ambiente de Experimentação

Os experimentos apresentados nesta dissertação foram executados em um computador com as seguintes especificações:

- Processador Intel Core i7-9700 com 8 núcleos (3.2 GHz e 8 MB Cache);
- 16GB DDR4 de memória RAM 3200mhz (2 pentes de 8GB);
- Placa de vídeo (GPU) NVIDIA GeForce GTX 1660 Ti com 6 GB GDDR6 de memória com arquitetura *Turing*, 1536 CUDA, 1770 MHz;

4.2 Desempenho do Rapid-DTW

Nesta seção são apresentados experimentos que visam avaliar o desempenho do Rapid-DTW em relação a outras soluções presentes na literatura. Foram realizadas comparações com os algoritmos P-TWDTW e a versão do TWDTW na linguagem C, apresentadas em [48, 47]. Os algoritmos foram executados 10 vezes em cada experimento realizado. Os resultados apresentados são a média aritmética das 10 execuções. Também foi realizada uma análise estatística dos resultados, apresentando os desvios padrões e intervalos de confiança com 95% de confiança, normal e t de student.

Tendo em vista que o resultado final dos algoritmos é o mesmo (i.e., solução exata via programação dinâmica), e o impacto no tempo de execução da computação da MPD está relacionado a quantidade de observações das séries temporais (i.e., o n da complexidade quadrática do problema representa o tamanho das séries temporais e padrões de entrada), para realizar experimentos com séries temporais de diferentes tamanhos, foram gerados dados de entrada de forma sintética a partir da base de dados MOD13Q1 apresentada em [46, 48].

Inicialmente, foram realizados testes para avaliar o desempenho do Rapid-DTW em conjuntos de dados com tamanhos frequentemente utilizados na literatura para classificação através de dados MOD13Q1 [56, 21, 39]. Nestes testes, foram gerados 50 padrões e 1000 séries temporais, com 24 observações para os padrões e 50 observações para as séries temporais, visando avaliar o tempo de execução da computação da matriz MPD neste cenário. Em seguida, para realizar o experimento com séries temporais maiores, tendo como objetivo demonstrar a capacidade do Rapid-DTW de processar séries temporais das próximas gerações de satélites, foi gerado outro conjunto de dados, com 50 padrões e 1000 séries temporais de tamanhos variando de 48 a 768 observações para padrões e 100 a 1600 observações para as séries temporais. Neste experimento são avaliados os tempos de execução completos dos algoritmos TWDTW em C++, P-TWDTW e Rapid-DTW na geração de meta-características.

A complexidade quadrática de espaço e tempo de algoritmos baseados no DTW quando combinada com grandes valores de entrada para n faz com que o tempo de execução e a quantidade de memória necessária para a computação das matrizes sejam muito elevados. Diante deste desafio, um último conjunto de dados foi criado para testar padrões e séries temporais simulando sensores espaciais com resolução temporal extremamente alta. Para este experimento, apenas 5 padrões e 10 séries temporais foram gerados, com tamanhos variando de 1536 a 12288 observações para padrões e 3200 a 24800 observações para as séries temporais. O P-TWDTW não foi utilizado neste último conjunto de dados, devido à limitação de escalabilidade.

O Rapid-DTW realiza a computação da MPD em janelas de elementos que podem ter seu tamanho adaptado de acordo com o tamanho de entradas dos dados. Assim, durante a realização dos experimentos os tamanhos de janela são alterados entre 2 e 384. Foi utilizado nos dois últimos experimentos testes com um padrão de configuração para o tamanho da janela buscando sempre ter blocos com 32 *threads*. Blocos de tamanhos múltiplos de 32 também apresentam melhor desempenho, devido as instruções em *warps* definidas pela arquitetura [44, 15]. Desta forma, os experimentos avaliam tanto o desempenho quanto a capacidade de adaptação dos algoritmos em relação ao conjunto de dados.

4.2.1 Resultados e discussões

A primeira etapa de experimentos foi realizada com 50 padrões e 1000 séries temporais, com seus tamanhos fixados em 24 para o padrão e 50 para as séries temporais. A Figura 4.1 apresenta um gráfico com os resultados de tempo de execução desta etapa, com a comparação entre os tempo de execução da MPD pelos algoritmos P-TWDTW, e Rapid-DTW com diferentes tamanhos de janelas de elementos. Na Tabela 4.1 são apresentados os tempos das execuções em milissegundos, juntamente com os dados estatísticos.

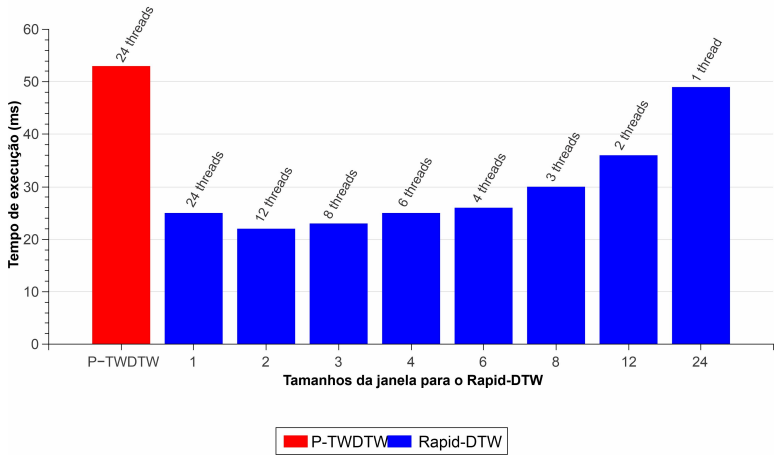


Figura 4.1: Tempo de execução da MPD com o Rapid-DTW e P-TWDTW. Padrões de tamanho 24 e séries temporais de tamanho 50.

Resultados e Dados Estatísticos														
Tamanho da janela	**Ex1**	**Ex1**	**Ex3**	**Ex4**	**Ex5**	**Ex6**	**Ex7**	**Ex8**	**Ex9**	**Ex10**	**Média**	**Desvio padrão**	**Intervalo de confiança normal**	**Intervalo de confiança T de Student**
P-TWDTW(1)	52	53	53	53	52	54	53	53	53	53	**52.9**	0.57	0.35	0.41
2	22	22	22	22	23	22	23	23	22	22	**22.3**	0.48	0.30	0.35
3	22	23	23	23	23	23	23	23	24	23	**23**	0.47	0.29	0.34
4	24	24	25	25	25	25	25	25	25	25	**24.8**	0.42	0.26	0.30
6	24	25	25	25	24	26	27	26	25	25	**25.2**	0.92	0.57	0.66
8	29	29	29	30	30	30	30	30	30	30	**29.7**	0.48	0.30	0.35
12	36	36	36	35	36	36	37	36	36	36	**36**	0.47	0.29	0.34
24	48	48	48	48	49	49	50	49	49	51	**48.9**	0.99	0.62	0.71

Tabela 4.1: Tempos de execução e dados estatísticos em relação a execução da computação da MPD, para padrões de tamanho 24 e séries temporais de tamanho 50

Observa-se que o Rapid-DTW chegou a ter um tempo de resposta $2,4$ vezes menor do que o P-TWDTW com a janela de tamanho 2, que é o menor múltiplo de 24. A medida em que o tamanho das janelas aumenta, a diferença entre os tempos de execução diminui. Isto ocorreu devido ao número de *threads* por bloco neste experimento ser menor do que 32. Como arquitetura CUDA faz uso de *warps*, conjuntos mínimos de 32 *threads* que compartilham instruções e memória, diminuir número de *threads* neste cenário tem

um peso maior do que o ganho com a diminuição da ociosidade de processamento. Ainda assim, é possível observar que o Rapid-DTW obteve melhor desempenho em todos os experimentos. De acordo com os dados estatísticos, os valores de desvio padrão são muito baixos, resultando em intervalos de confiança pequenos. Estes resultados geram maior confiabilidade em relação a otimização de performance apresentada pelo Rapid-DTW. Todas as tabelas apresentadas com dados estatísticos apresentam o mesmo padrão nos resultados.

Em problemas de complexidade quadrática, a medida que o tamanho da entrada de dados aumenta, o tempo de execução tende a ser significativamente maior. Na segunda estratégia de experimentação, o comportamento dos algoritmos foi verificado com diversos tamanhos de padrões e séries temporais. Neste experimento são avaliados os tempos de execução das 3 soluções em avaliação realizando todas as etapas para computação das meta-características, com os tamanhos de padrões variando de 48 a 768 e tamanhos variando de séries temporais de 100 a 1600. Os resultados destes testes estão ilustrados no gráfico apresentado na Figura 4.2. A Tabela 4.2 apresenta os tempos de execução e dados estatísticos.

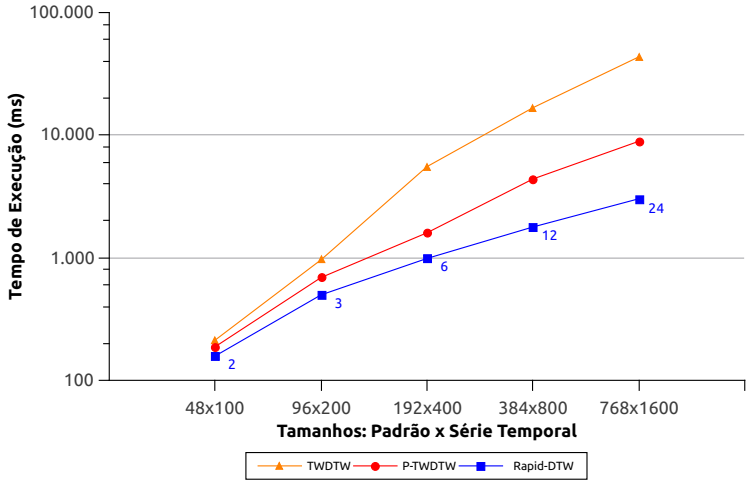


Figura 4.2: Comparação de tempos de execução completos para geração de meta-características dos algoritmos avaliados. Os números em azul, próximos aos resultados do Rapid-DTW, representam o tamanho da janela utilizada no Rapid-DTW, variando entre 2 e 24. Eixo *y* em escala logarítmica.

Diante destes resultados, é importante observar que a medida em que aumentamos o tamanho da entrada do problema, o desempenho do Rapid-DTW em relação ao P-TWDTW melhora. Isto ocorre devido a possibilidade de adequação do tamanho da janela ao tamanho das entradas, pois ao aumentarmos o tamanho da janela, diminuímos a quantidade de *threads* ociosas durante a execução, permitindo a obtenção de melhores resultados em conjuntos de dados maiores. Os melhores tamanhos de janelas pode ser obtidos de forma empírica, ou seja, através de experimentação com diversos tamanhos.

Resultados Rapid-DTW														
Entrada de dados	**Ex1**	**Ex2**	**Ex3**	**Ex4**	**Ex5**	**Ex6**	**Ex7**	**Ex8**	**Ex9**	**Ex10**	**Média**	**Desvio padrão**	**Intervalo de confiança normal**	**Intervalo de confiança T de Student**
48x100	154	159	160	158	161	152	161	158	158	159	**158**	2.91	1.80	2.08
96x200	496	499	492	501	502	498	499	495	499	500	**498.1**	3.00	1.86	2.14
192x400	988	991	985	987	986	988	985	988	990	985	**987.3**	2.11	1.31	1.51
384x800	1788	1784	1780	1788	1780	1781	1782	1777	1781	1782	**1782.3**	3.50	2.17	2.50
768x1600	3025	3025	3027	3030	3033	3029	3024	3040	3028	3022	**3028.3**	5.21	3.23	3.73
Resultados P-TWDTW														
Entrada de dados	**Ex1**	**Ex2**	**Ex3**	**Ex4**	**Ex5**	**Ex6**	**Ex7**	**Ex8**	**Ex9**	**Ex10**	**Média**	**Desvio padrão**	**Intervalo de confiança normal**	**Intervalo de confiança T de Student**
48x100	185	185	185	189	191	192	190	189	188	188	**188.2**	2.53	1.57	1.81
96x200	682	685	684	686	689	690	694	690	690	690	**688**	3.62	2.24	2.59
192x400	1594	1597	1597	1597	1596	1599	1600	1602	1597	1599	**1597.8**	2.25	1.40	1.61
384x800	4385	4385	4386	4386	4387	4390	4392	4393	4393	4395	**4389.2**	3.82	2.37	2.74
768x1600	8910	8910	8910	8911	8911	8912	8912	8915	8914	8915	**8912**	2.00	1.24	1.43
Resultados TWDTW														
Entrada de dados	**Ex1**	**Ex2**	**Ex3**	**Ex4**	**Ex5**	**Ex6**	**Ex7**	**Ex8**	**Ex9**	**Ex10**	**Média**	**Desvio padrão**	**Intervalo de confiança normal**	**Intervalo de confiança T de Student**
48x100	210	212	210	213	215	214	214	213	215	213	**212.9**	1.79	1.11	1.28
96x200	959	961	964	963	963	963	963	964	965	964	**962.9**	1.73	1.07	1.24
192x400	5536	5538	5538	5538	5538	5538	5539	5538	5538	5539	**5538**	0.82	0.51	0.58
384x800	16736	16736	16738	16738	16738	16737	16739	16741	16738	16739	**16738**	1.49	0.92	1.07
768x1600	43516	43518	43517	43519	43519	43519	43521	43522	43524	43526	**43520.1**	3.14	1.95	2.25

Tabela 4.2: Resultados e dados estatísticos do experimento de comparação entre os tempos de execução do Rapid-DTW, P-TWDTW e TWDTW. Os tempos das execuções estão em representados em milissegundos.

Para entender melhor a otimização do uso de *threads* pode-se abstrair destes testes a porcentagem de iterações em que todas as *threads* trabalham de forma simultânea durante a computação da MPD. Esse dado é obtido através da razão entre a quantidade de iterações com todas as *threads* executando em paralelo e o total de iterações para computação de toda a MPD. Este dado é ilustrado no gráfico da Figura 4.3. Observa-se que o P-TWDTW possui sempre aproximadamente 35,5% de iterações com todas as *threads* trabalhando simultaneamente em quaisquer conjuntos de dados, enquanto o Rapid-DTW chega a ter 96,1% das iterações com todas as *threads* trabalhando no conjunto de entrada 768×1600 e janela de tamanho 24.

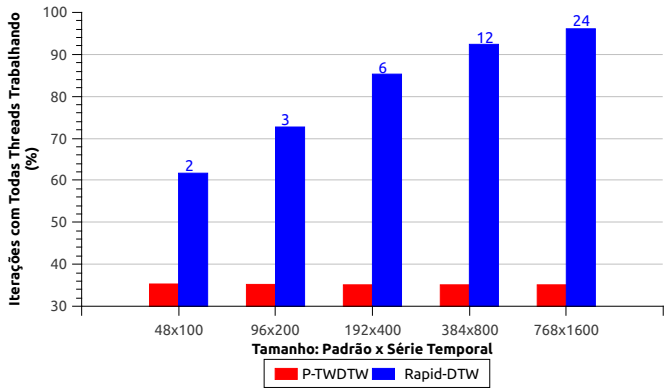


Figura 4.3: Porcentagem de iterações com todas as *threads* trabalhando em paralelo durante a computação da MPD.

Na terceira etapa de experimentos, foram aplicados dados que representam produtos com resoluções temporais que ultrapassam a capacidade máxima de processamento

do P-TWDTW, variando os tamanhos de padrões de 1536 a 12288 e variando os tamanhos de séries temporais de 3200 a 24800. A Figura 4.4 apresenta o gráfico destes experimentos, sem a presença do P-TWDTW devido as limitações de escalabilidade. Como o cálculo de *speedup* de uma solução paralela é obtido através da comparação com a melhor solução sequencial, o gráfico da Figura 4.5 apresenta o *speedup* do Rapid-DTW em relação ao TWDTW implementado na linguagem C++. A Tabela 4.3 apresenta os tempos de execução e dados estatísticos.

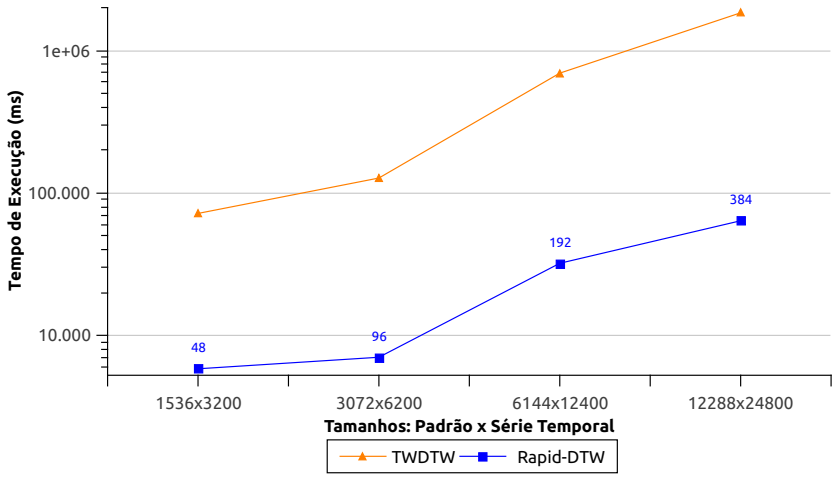


Figura 4.4: Comparação de tempos de execução completos para geração de meta-características do TWDTW em C++ e do Rapid-DTW. Os números em azul representam o tamanho da janela utilizado no Rapid-DTW, variando entre 48 e 384. Eixo y em escala logarítmica.

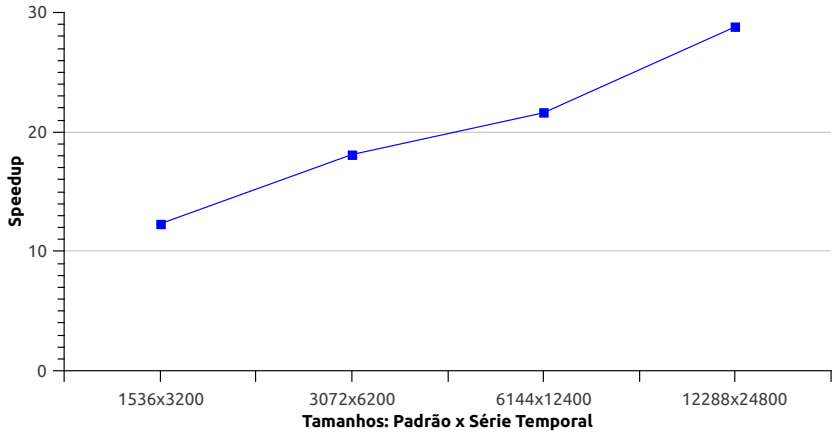


Figura 4.5: Speedup do Rapid-DTW em relação a versão sequencial TWDTW em C++.

Observa-se nestes resultados, que o padrão da melhora no desempenho a medida em que o tamanho dos dados de entrada aumentam, continua a ocorrer nos testes que

Resultados Rapid-DTW													
Entrada de dados	**Ex1**	**Ex1**	**Ex3**	**Ex4**	**Ex5**	**Ex6**	**Ex7**	**Ex8**	**Ex9**	**Ex10**	**Média**	**Desvio padrão**	**Intervalo de confiança normal**	**Intervalo de confiança T de Student**
1536x3200	5832	5832	5840	5838	5836	5842	5839	5849	5839	5841	**5838.8**	4.96	3.08	3.55
3072x6200	7035	7033	7041	7034	7037	7039	7039	7037	7037	7035	**7036.7**	2.50	1.55	1.79
6144x12400	32117	32102	32118	32122	32119	32120	32120	32119	32118	32117	**32117.2**	5.55	3.44	3.97
12288x24800	63852	63859	63856	63859	63866	63861	63867	63862	63864	63866	**63861.2**	4.83	2.99	3.45
Resultados TWDTW														
Entrada de dados	**Ex1**	**Ex1**	**Ex3**	**Ex4**	**Ex5**	**Ex6**	**Ex7**	**Ex8**	**Ex9**	**Ex10**	**Média**	**Desvio padrão**	**Intervalo de confiança normal**	**Intervalo de confiança T de Student**
1536x3200	71888	71890	71889	71899	71898	71899	71889	71889	71893	71895	**71892.9**	4.51	2.79	3.22
3072x6200	127380	127384	127382	127385	127388	127388	127379	127377	127371	127377	**127381.1**	5.38	3.34	3.85
6144x12400	693732	693736	693739	693738	693741	693740	693746	693739	693741	693745	**693739.7**	4.06	2.51	2.90
12288x24800	1839195	1839188	1839200	1839210	1839214	1839212	1839201	1839203	1839205	1839206	**1839203.4**	7.92	4.91	5.66

Tabela 4.3: Resultados e dados estatísticos da comparação entre o Rapid-DTW e o algoritmo sequencial TWDTW.

envolvem todas as etapas para computação de meta-características. Nos testes em que foi possível comparar o Rapid-DTW com o P-TWDTW, o maior ganho de desempenho foi obtido no maior conjunto de testes, alcançando um tempo de execução $2,9$ vezes menor. Em comparação com a versão sequencial, o Rapid-DTW obteve um *speedup* de até $28,8$ vezes no maior conjunto de dados avaliado. Isto acontece devido ao acréscimo de tempo execução da MPD inerente ao aumento do tamanho dos dados de padrões e séries temporais. Neste caso, o tempo de computação da MPD é significativamente maior do que o restante das etapas, chegando a 87% do tempo de execução.

4.3 Efetividade das meta-características na classificação do uso e cobertura do solo com Random Forest

As meta-características geradas pelo Rapid-DTW podem ser utilizadas em diversas análises, entre elas, a classificação do uso e cobertura do solo. O trabalho de [40] propõe o uso do algoritmo TWDTW em conjunto com o k-NN, em uma abordagem de classificação do uso e cobertura do solo baseado em medidas de similaridade de STSR. Já o trabalho de [46] utiliza estas meta-características como um complemento á metodologia de classificação baseada no classificador SVM proposta em [56]. Nesta seção são apresentados diversos cenários de aplicação das meta-características geradas pelo Rapid-DTW em modelos de classificação do uso e cobertura do solo, utilizando como base para desenvolvimento destes, o classificador Random Forest, tendo em vista sua popularidade na área de sensoriamento remoto, aliada a bons resultados de acurácia. Não foram encontrados trabalhos anteriores que avaliem a aplicação de meta-características baseadas em similaridade de STSR geradas a partir do conjunto das metodologias DTW e Time-Weighted com o classificador Random Forest. A finalidade destes experimentos é obter uma resposta em relação a utilização das meta-características para otimizar a acurácia de modelos de classificação do uso e cobertura do solo.

4.3.1 Cenários de análise e resultados

Para avaliar as meta-características geradas pelo Rapid-DTW, este trabalho utilizou como base a metodologia de classificação do uso e cobertura do solo apresentada em [56], em que os valores de observações de cada banda são utilizados sem transformações na composição do espaço de características. Para fins de comparação, o mesmo conjunto de dados foi utilizado. São dados MOD13Q1, com 23 observações anuais das bandas *Near-infrared* (nir), *Mid-infrared* (mir) e dos índices *Normalized Difference Vegetation Index* (NDVI), *Enhanced Vegetation Index* (EVI). Os dados foram extraídos da região do Mato Grosso - Brasil, com o objetivo de especificar nove classes diferentes de uso e cobertura do solo, distribuídos de acordo com a Tabela 4.4. A complexidade de classificar este conjunto de dados é elevada, devido ao comportamento similar das séries temporais associadas a agricultura [56, 46]. A semelhança no comportamento destas séries temporais pode ser observada na Figura 4.6.

Classe	Amostras
1 - Cerrado	400
2 - Algodão-pousio	34
3 - Floresta	138
4 - Pastagem	370
5 - Soja-milho	398
6 - Soja-algodão	399
7 - Soja-pousio	88
8 - Soja-milheto	235
9 - Soja-girassol	53
Total	2115

Tabela 4.4: Distribuição de amostras entre as classes.

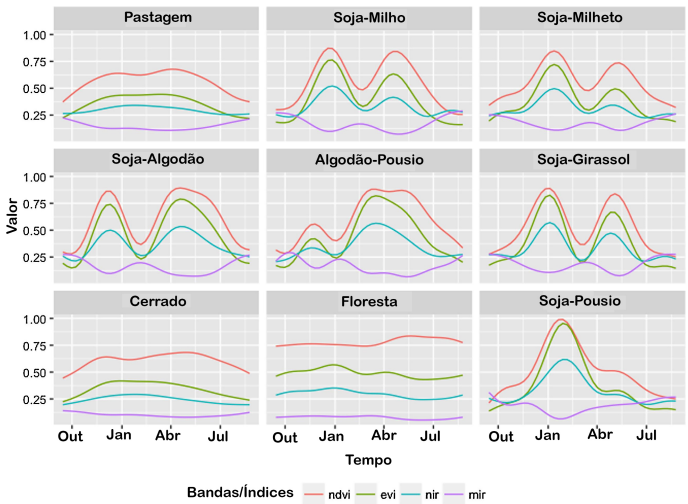


Figura 4.6: Comportamento das séries temporais das classes analisadas [56].

Os dados de padrões de comportamento das séries temporais foram obtidos através do trabalho de [46], onde foram gerados utilizando o método GAM [67]. Foram geradas meta-características para cada ponto das amostras utilizando o Rapid-DTW. As medidas de similaridade representam o quão próximo é o comportamento de cada série temporal com cada um dos nove padrões analisados. Portanto, para cada pixel analisado, há 92 características (i.e, duas bandas e dois índices de 23 observações por pixel) e 9 meta-características, representando a distância de cada pixel para uma classe específica. A Tabela 4.5 apresenta um exemplo resumido de um espaço de características possuindo 3 pixels com 2 observações para cada banda e índice, e 3 meta-características.

Pixel	ndvi1	ndvi2	evi1	evi2	nir1	nir2	mir1	mir2	meta-característica Cerrado	meta-característica Algodão-pousio	meta-característica Floresta
1	0,39	0,49	0,25	0,28	0,32	0,27	0,31	0,17	3,53	4,03	4,00
2	0,50	0,49	0,26	0,33	0,23	0,36	0,14	0,16	2,62	4,27	2,91
3	0,35	0,34	0,19	0,16	0,23	0,20	0,20	0,15	1,70	4,08	3,34

Tabela 4.5: Exemplo de espaço de características

Para analisar e avaliar o impacto da utilização de meta-características baseadas em similaridade geradas a partir do algoritmo Rapid-DTW, foram criados diferentes espaços de características. Foi desenvolvida uma metodologia que se inicia com a seleção de diversos espaços características compostos de forma distintas. Com os espaços de características consolidados, são realizados os treinamentos e a validação dos classificadores, finalizando com uma etapa de análise e avaliação dos resultados.

Durante a etapa de classificação e validação os classificadores Random Forest foram treinados com 500 árvores, pois não foi possível observar melhora de resultados com o aumento do número de árvores. Todos os experimentos foram realizados utilizando a técnica de validação cruzada K-Fold, com K = 5 [68, 56]. Para facilitar a análise foram geradas matrizes de confusão com as acurácias do produtor, do usuário e global. A acurácia do produtor permite a analise da capacidade de acerto em relação a uma classe específica enquanto acurácia do usuário permite analisar a quantidade de amostras de uma classe especifica que foram mapeadas para outra. A acurácia global corresponde a quantidade total de acertos em relação ao conjunto de amostras. Além disto, o índice Kappa também foi computado, visando apresentar um resultado baseado no grau de concordância da classificação [56].

Um fluxograma da metodologia utilizada para realização da avaliação é apresentado na Figura 4.7. Através desta metodologia comparou-se o impacto das meta-características em diferentes abordagens. A codificação das metodologias de classificação do uso e cobertura do solo apresentadas nessa dissertação foi realizada utilizando a linguagem Python [1], através da biblioteca de aprendizado de máquina scikit-learn [53].

[1] https://www.python.org/

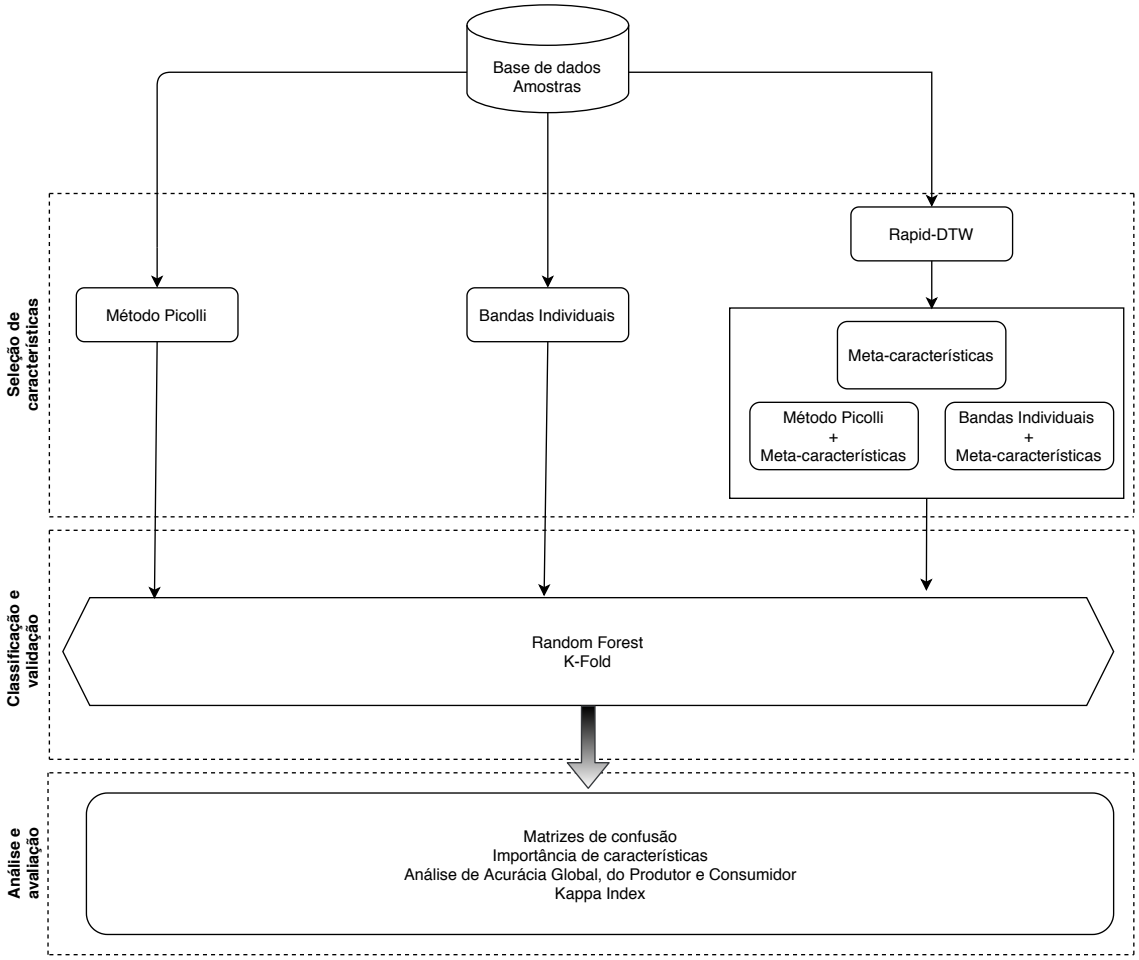


Figura 4.7: Fluxograma da metodologia utilizada para a analisar as meta-características.

Cenário de comparação A

Este cenário visa analisar a abordagem apresentada por [56] aplicada ao classificador Random Forest em conjunto com as meta-características. Em um primeiro experimento o classificador é treinado com um espaço de características que contem todas as 23 observações anuais das bandas NIR e MIR e dos índices NDVI e EVI, totalizando um espaço de características de tamanho 92. Em seguida, um segundo experimento é realizado, acrescentando a este espaço de característicos as 9 meta-características geradas pelo Rapid-DTW, totalizando 101 características por pixel. Diante dos resultados, são analisadas as matrizes de confusão dos 2 experimentos, com a finalidade de observar pontos em que a adição das meta-características tiveram impacto positivo ou negativo. Neste cenário também foi realizada uma análise de importância de características, permitindo identificar a relevância das meta-características geradas pelo Rapid-DTW na classificação do uso e cobertura do solo com Random Forest em um grande espaço de características.

O resultado do primeiro experimento deste cenário podem ser observados na Tabela 4.6. Neste experimento o classificador Random Forest foi treinado com 23 obser-

vações das bandas (NIR, MIR) e índices (EVI, NDVI), totalizando de 92 características por pixel. Observa-se que o classificador obteve uma acurácia global de 92,48%, o índice Kappa deste experimento foi de 0.91. Neste experimento o classificador conseguiu boa acurácia (i.e., maior que 90% de acurácia do produtor) para as classes Cerrado, Floresta, Pastagem, Soja-milho, Soja-algodão e Soja-pousio. Houve bastante confusão entre as classes Soja-Girassol e Soja-Milho devido ao comportamento similar das séries temporais dessas classes. O mesmo ocorreu entre as classes Soja-milho e Soja-milheto, fazendo com que a classe Soja-milheto também ficasse com a acurácia do produtor abaixo de 90%. O comportamento semelhante das classes Algodão-pousio e Soja-algodão prejudicou o desempenho da classificação da classe Algodão-pousio, fazendo com que ela obtivesse apenas um 73,53% de Acurácia do produtor.

	1	2	3	4	5	6	7	8	9	**AU (%)**
1 - Cerrado	**398**	0	1	8	1	0	0	0	0	97,50%
2 - Algodão-pousio	0	**25**	0	0	0	2	0	0	0	94,12%
3 - Floresta	2	0	**136**	1	0	0	0	0	0	97,83%
4 - Pastagem	0	3	1	**359**	5	1	0	9	0	94,86%
5 - Soja-milho	0	0	0	0	**360**	26	0	27	16	82,66%
6 - Soja-algodão	0	6	0	1	8	**366**	0	3	0	95,49%
7 - Soja-pousio	0	0	0	0	0	0	**87**	1	0	98,86%
8 - Soja-milheto	0	0	0	1	23	4	1	**195**	7	84,68%
9 - Soja-girasol	0	0	0	0	1	0	0	0	**30**	98,11%
AP (%)	99,50%	73,53%	98,55%	97,03%	90,45%	91,73%	98,86%	82,98%	56,60%	**AG = 92,48%**

Tabela 4.6: Matriz de confusão do Método Picolli

	1	2	3	4	5	6	7	8	9	**AU(%)**
1 - Cerrado	**396**	0	1	6	0	0	0	0	0	98,25%
2 - Algodão-pousio	0	**28**	0	0	0	1	0	0	0	97,06%
3 - Floresta	2	0	**136**	1	0	0	0	0	0	97,83%
4 - Pastagem	2	1	1	**361**	5	1	0	6	0	95,68%
5 - Soja-milho	0	0	0	0	**360**	23	0	26	14	84,17%
6 - Soja-algodão	0	4	0	1	8	**369**	0	2	0	96,24%
7 - Soja-pousio	0	0	0	1	0	1	**86**	1	0	96,59%
8 - Soja-milheto	0	1	0	0	24	4	2	**200**	7	83,83%
9 - Soja-girassol	0	0	0	0	1	0	0	0	**32**	98,11%
AP (%)	99,00%	82,35%	98,55%	97,57%	90,45%	92,48%	97,73%	85,11%	60,38%	**AG = 93,05%**

Tabela 4.7: Matriz de confusão do Método Picolli + meta-características

Ao adicionar as 9 meta-características por pixel, gerando um espaço de características de tamanho 101, os resultados de acurácia obtiveram melhorias. Estes resultados podem ser observados na Tabela 4.7. Neste experimento a acurácia global obtida foi de 93,05% e índice Kappa 0.92. Neste cenário, a contribuição das meta-características produzidas pelo Rapid-DTW foi mais significativa nas classes difíceis de classificar (i.e, classes que, por possuírem comportamento semelhantes nas séries temporais, acabam confundindo o classificador). Traçando um comparativo entre os experimentos realizados, tivemos um aumento da acurácia do produtor de 8,82% na classe Algodão-pousio, 2,13% em Soja-milheto e 3,78% em Soja-girassol. Estes resultados mostram que as meta-

características podem contribuir para a classificação de classes complexas, sendo capazes de aumentar a acurácia em diferentes classes de agricultura.

Para avaliar o impacto das meta-características durante este experimento de classificação, foi utilizada uma função que ranqueia a importância de cada característica durantes a classificação realizada pelo Random Forest. Esta análise permite identificar que a importância dessas 9 meta-características por pixel é significativamente maior quando comparada com os dados puros de bandas e Índices. A Figura 4.8 apresenta o gráfico com a relação da importância de características. Diante dos resultados, conclui-se que as meta-características possuem uma boa representação das amostras, tornado-as mais importantes durante a classificação. Neste cenário, as meta-características foram capazes de aumentar o resultado de acurácia de uma metodologia que já apresentava altos valores de acurácia.

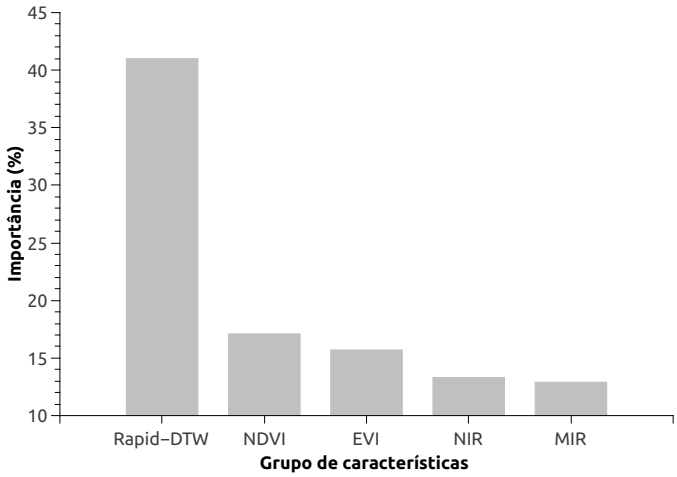


Figura 4.8: Importância das características na classificação

Cenário de comparação B

Neste cenário são analisadas situações em que o espaço de características é menor, com baixos valores de acurácia. Para isto, foram realizados experimentos com as bandas e índices individualmente, treinando 4 classificadores, com espaços de características compostos apenas pelas 23 observações por pixel, um classificador para cada dado disponível (NDVI, EVI, NIR e MIR). Em seguida, os treinamentos são realizados novamente, com a adição das 9 meta-características para cada pixel, agora com espaços de características de tamanho 32. Para estes experimentos foram comparadas apenas as acurácias globais na classificação das amostras para cada classificador, os resultados podem ser observados na Tabela 4.8.

Com o espaço de características reduzido (i.e. usando apenas as 23 observações de uma Banda ou Índice, somados a 9 meta-características geradas pelo Rapid-DTW), foi

	Bandas Individuais	**Bandas Individuais + meta-características**
1 - NIR	AG = 86,8% Kappa = 0.84	AG = 90,35% Kappa = 0.88
2 - MIR	AG = 83,26% Kappa = 0.80	AG = 89,17% Kappa = 0.87
3 - EVI	AG = 87,47% Kappa = 0.85	AG = 90,3% Kappa = 0.88
4 - NDVI	AG = 86,43% Kappa = 0.84	AG = 90,4% Kappa = 0.88

Tabela 4.8: Comparação entre experimentos Bandas Individuais e Bandas Individuais + meta-características

possível observar uma melhora de até 3,97% na acurácia global no modelo com NDVI, alcançando a maior acurácia global deste cenário de testes, com 90,40%. O experimento que mais se beneficiou com a adição das meta-características foi o que possuía a menor acurácia global, utilizando as observações da banda MIR, neste modelo foi possível obter uma melhora de até 5,91%. Estes resultados demonstram que estas meta-características podem ser aplicadas em diversos modelos de classificação que utilizem o Random Forest, podendo levar a melhoras na acurácia.

Cenário de comparação C

Todos os cenários anteriores utilizaram como base para classificação o Método Picoli, utilizando como características os dados de observação das séries temporais sem nenhuma transformação e em seguida avaliando os mesmos experimentos com a adição das meta-características geradas pelo Rapid-DTW. Neste cenário de comparação é proposta uma abordagem diferente. O experimento é realizado utilizadas apenas as 9 meta-características baseadas em similaridade geradas pelo Rapid-DTW como entrada para o algoritmo Random Forest. Esta estratégia tem como objetivo a comparação com a metologia de classificação do uso de cobertura do solo utilizando o TWDTW em conjunto com o k-NN [40, 46].

A Tabela 4.9 apresenta a matriz de confusão com os resultados deste experimento. Observa-se que essa abordagem atingiu uma acurácia global de 84,02% e Kappa 0.81, utilizando apenas as 9 meta-características geradas pelo Rapid-DTW. O mesmo experimento, utilizando o mesmo conjunto de dados foi realizado no trabalho de [46], no qual foi proposta uma combinação das meta-características geradas pelo P-TWDTW com o k-NN, obtendo acurácia global de 78%. Neste caso, o Random Forest conseguiu obter um melhor resultado, em termos de acurácia, de 6,2% em relação ao k-NN, sem alterações no espaço de características.

Pode-se observar que os valores de acurácia do produtor se mantiveram acima de 90% para as classes Cerrado, Floresta, Pastagem e Soja-Pousio. Em contraste, o restante

	1	2	3	4	5	6	7	8	9	**AU(%)**
1 Cerrado	**375**	0	7	16	0	0	0	0	0	94,25%
2 Algodão-pousio	0	**13**	0	0	0	5	0	0	0	85,29%
3 Floresta	11	0	**126**	4	0	0	0	0	0	89,13%
4 Pastagem	14	0	5	**343**	2	0	0	7	0	92,43%
5 Soja-milho	0	2	0	1	**326**	48	0	44	35	67,34%
6 Soja-algodão	0	17	0	1	22	**333**	0	6	2	87,97%
7 Soja-pousio	0	1	0	0	0	1	**81**	7	0	89,77%
8 Soja-milheto	0	1	0	5	46	12	7	**171**	7	66,81%
9 Soja-Girassol	0	0	0	0	2	0	0	0	**9**	96,23%
AP(%)	93,75%	38,24%	91,30%	92,70%	81,91%	83,46%	92,05%	72,77%	16,98%	**AG = 84,02%**

Tabela 4.9: Matriz de confusão da classificação utilizando apenas as 9 meta-características.

das classes, relacionadas a cultivo de agriculturas, obtiveram piores resultados. Isto ocorre porque estas classes apresentam comportamentos semelhantes em suas séries temporais (Figura 4.6), dificultando para classificador sua diferenciação, e consequentemente obter bons resultados de acurácia [56, 49].

Por fim, dadas as semelhanças entre as classes de agricultura, um nova abordagem do experimento foi conduzida unificando as classes Algodão-Pousio, Soja-milho, Soja-algodão, Soja-pousio, Soja-algodão, Soja-milheto e Soja-Girassol em uma única classe chamada Agricultura. Nesta abordagem foi possível observar melhores resultados de acurácia. O classificador conseguiu obter 96,50% de acurácia global. Para a nova classe de Agricultura, obteve-se uma acurácia do produtor de 99,59%. Com estes resultados, em cenários onde não há necessidade de tipificar a classificação da agricultura, a utilização das meta-características pode se tornar uma abordagem competitiva. A Tabela 4.10 apresenta a matriz de confusão deste experimento.

	1	2	3	4	**AU (%)**
1 Cerrado	**375**	7	16	0	94,25%
2 Floresta	11	**127**	3	0	89,86%
3 Pastagem	14	4	**337**	5	93,78%
4 Agricultura	0	0	14	**1202**	98,84%
AP(%)	93,75%	92,03%	91,08%	99,59%	**AG = 96,50%**

Tabela 4.10: Matriz de confusão da classificação utilizando apenas as 9 meta-características com simplificação das classes de agricultura.

CAPÍTULO 5

Conclusões

Diante do grande número de satélites sendo constantemente lançados na órbita da Terra, e seus sensores cada vez mais poderosos, espera-se que as resoluções temporais e espaciais das STSR continuem a aumentar no futuro. Portanto, o custo computacional para processar esse grande volume de dados também deve aumentar proporcionalmente. Neste cenário, o futuro do Sensoriamento Remoto pode depender da exploração de ferramentas de computação de alto desempenho, fazendo uso de processamento paralelo [30, 29, 57, 9].

Nesta dissertação foi apresentado um algoritmo paralelo para a geração de meta-características baseadas em similaridade de STSR. Estas meta-características podem ser utilizadas principalmente na composição de espaços de características para classificação automatizada do uso e cobertura do solo. Um espaço de características que representem bem os elementos a serem classificados é crucial para uma boa acurácia do classificador [4]. As meta-características geradas pelo Rapid-DTW são um compilado de informações de toda a série temporal de bandas e índices. Por este motivo, estas meta-características correspondem a uma ótima representação das classes, causando um impacto positivo na acurácia da classificação.

O Rapid-DTW explora paralelismo para lidar com o grande volume de dados de sensoriamento remoto, demonstrando o potencial das estratégias de processamento paralelo. Os experimentos realizados mostraram que o Rapid-DTW apresenta melhorias significativas de desempenho em comparação com os métodos tradicionais presentes da literatura. Além disto, este trabalho propõe a aplicação das meta-características em modelos de classificação baseados no classificador Random Forest. Esta abordagem mostrou ser possível obter melhora de acurácia em diversos modelos com a adição das meta-características em seus espaços de características. Através destes resultados, foi possível elaborar estratégias que utilizam apenas as meta-características como entrada para o classificador, mantendo bons índices de acurácia. Apesar do alto custo computacional para a geração das meta-características, uma vez que elas sejam geradas pelo Rapid-DTW, pode-se utilizá-las em diversos modelos de classificação.

5.1 Contribuições

A principal contribuição desta dissertação é o algoritmo Rapid-DTW, uma solução baseada no método DTW eficiente para geração de meta-características baseadas em similaridade de STSR. O Rapid-DTW é o primeiro passo para o desenvolvimento de aplicações que possam gerar meta-características para diversos produtos de sensoriamento remoto. Assim como índices EVI e NDVI, as meta-características baseadas em similaridade podem se tornar um dado relevante no contexto de classificação do uso e cobertura do solo.

Este trabalho também apresentou a aplicação e avaliação das meta-características em diversos modelos de classificação baseados no Random Forest. Esta contribuição demonstra a possibilidade da utilização das meta-caraterísticas para a melhoria de modelos de classificação já consolidados.

5.2 Limitações

Apesar da obtenção de bons resultados, este trabalho possui algumas limitações. A estratégia da computação dos elementos em janelas faz com que o algoritmo necessite que a configuração do tamanho da janela seja múltiplo do menor valor de tamanho entre a série temporal e o padrão. Caso este valor seja um número primo, o tamanho da janela será necessariamente 1 ou o valor em si. Nestes casos, o algoritmo irá funcionar exatamente como o P-TWDTW, para tamanhos de janelas igual a 1, ou de forma sequencial para tamanhos de janela iguais ao menor valor entre o padrão e a série temporal.

Outra limitação do trabalho está na definição do tamanho ideal de janela de acordo com o tamanho de entrada de dados. Não é possível saber com exatidão qual o tamanho de janela ideal para uma entrada de dados específica sem a realização de testes. Foram obtidos bons resultados com tamanhos de janelas que buscam trabalhar em blocos de 32 threads, mas os resultados podem variar, melhorando ou piorando o desempenho, quando não é possível obter uma configuração em que o tamanhos dos blocos seja múltiplo de 32. Como o Rapid-DTW foi desenvolvido e testado utilizando apenas uma GPU, a solução atual não é capaz de automaticamente trabalhar em um cenário com múltiplas GPUs, necessitando para isto extensões no código.

5.3 Trabalhos Futuros

Em trabalhos futuros pretende-se estender o Rapid-DTW para operar em ambientes com múltiplas GPUS e até com clusters de GPUs, aumentando ainda mais o grau de paralelismo. Há também a intenção de utilizar o Rapid-DTW para geração de meta-

características em outros bancos de dados, permitindo que seja possível a utilização destas meta-características na composição de espaços de características em diversos modelos de classificação, inclusive modelos de classificação que abrangem todo território nacional.

Referências Bibliográficas

[1] AYALA-IZURIETA, J. E.; MÁRQUEZ, C. O.; GARCÍA, V. J.; RECALDE-MORENO, C. G.; RODRÍGUEZ-LLERENA, M. V.; DAMIÁN-CARRIÓN, D. A. **Land cover classification in an ecuadorian mountain geosystem using a random forest classifier, spectral vegetation indices, and ancillary geographic data**. *Geosciences*, 7(2):34, 2017.

[2] BALA, G.; CALDEIRA, K.; WICKETT, M.; PHILLIPS, T.; LOBELL, D.; DELIRE, C.; MIRIN, A. **Combined climate and carbon-cycle effects of large-scale deforestation**. *Proceedings of the National Academy of Sciences*, 104(16):6550–6555, 2007.

[3] BELGIU, M.; ZHOU, Y.; MARSHALL, M.; STEIN, A. **Dynamic time warping for crops mapping**. *The International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, 43:947–951, 2020.

[4] BELGIU, M.; DRĂGUŢ, L. **Random forest in remote sensing: A review of applications and future directions**. *ISPRS Journal of Photogrammetry and Remote Sensing*, 114:24–31, 2016.

[5] BERNDT, D. J.; CLIFFORD, J. **Using dynamic time warping to find patterns in time series.** In: *KDD workshop*, volume 10, p. 359–370. Seattle, WA, USA:, 1994.

[6] BLYTHE, D. **Rise of the graphics processor**. *Proceedings of the IEEE*, 96(5):761–778, 2008.

[7] BONAN, G. B. **Forests and climate change: forcings, feedbacks, and the climate benefits of forests**. *science*, 320(5882):1444–1449, 2008.

[8] BREIMAN, L. **Random forests**. *Machine learning*, 45(1):5–32, 2001.

[9] CAMARA, G.; ASSIS, L. F.; RIBEIRO, G.; FERREIRA, K. R.; LLAPA, E.; VINHAS, L. **Big earth observation data analytics**. In: *Proceedings of the 5th ACM SIGSPATIAL International Workshop on Analytics for Big Geospatial Data - BigSpatial '16*. ACM Press, 2016.

[10] CANOVAS-GARCIA, F.; ALONSO-SARRIA, F. **Optimal combination of classification algorithms and feature ranking methods for object-based classification of**

submeter resolution z/i-imaging dmc imagery. *Remote Sensing*, 7(4):4651–4677, 2015.

[11] CELIK, T. **Unsupervised change detection in satellite images using principal component analysis and *k*-means clustering**. *IEEE Geoscience and Remote Sensing Letters*, 6(4):772–776, 2009.

[12] CHEN, D.-K.; SU, H.-M.; YEW, P.-C. **The impact of synchronization and granularity on parallel systems**. *ACM SIGARCH Computer Architecture News*, 18(2SI):239–248, 1990.

[13] CHI, M.; PLAZA, A.; BENEDIKTSSON, J. A.; SUN, Z.; SHEN, J.; ZHU, Y. **Big data for remote sensing: Challenges and opportunities**. *Proceedings of the IEEE*, 104(11):2207–2219, 2016.

[14] COSTA, B.; FREIRE, J.; CAVALCANTE, H.; HOMCI, M.; CASTRO, A.; VIÉGAS JR, R.; MEIGUINS, B.; MORAIS, J. **Fault classification on transmission lines using knn-dtw**. p. 174–187, 07 2017.

[15] CUDATM, N. **Nvidia cuda c programming best practices guide**. *NVIDIA Corporation*, 2701, 2009.

[16] DADI, M. M. **Assessing the transferability of random forset and time-weighted dynamic time warping for agriculture mapping**. Master's thesis, University of Twente, 2019.

[17] DANIELSSON, P.-E. **Euclidean distance mapping**. *Computer Graphics and image processing*, 14(3):227–248, 1980.

[18] DE OLIVEIRA, S. S. T.; DO SACRAMENTO RODRIGUES, V. J.; FERREIRA, L. G.; MARTINS, W. S. **P-twdtw: Parallel processing of time series remote sensing images using manycore architectures**. In: *2018 Symposium on High Performance Computing Systems (WSCAD)*, p. 252–258. IEEE, 2018.

[19] DE OLIVEIRA, S. S. T.; MARTINS, W. S.; SACRAMENTO, V.; BUENO, E.; CARDOSO, M.; PASCOAL, L. **A parallel and distributed approach to the analysis of time series on remote sensing big data**. *Journal of Information and Data Management*, 10(1):16–34, 2019.

[20] DEWAN, A. M.; YAMAGUCHI, Y. **Land use and land cover change in greater dhaka, bangladesh: Using remote sensing to promote sustainable urbanization**. *Applied geography*, 29(3):390–401, 2009.

[21] DONG, Q.; CHEN, X.; CHEN, J.; ZHANG, C.; LIU, L.; CAO, X.; ZANG, Y.; ZHU, X.; CUI, X. **Mapping winter wheat in north china using sentinel 2a/b data: A method based on phenology-time weighted dynamic time warping**. *Remote Sensing*, 12(8):1274, 2020.

[22] DORN, F. **Dynamic programming and fast matrix multiplication**. In: *Lecture Notes in Computer Science*, p. 280–291. Springer Berlin Heidelberg, 2006.

[23] DURO, D. C.; FRANKLIN, S. E.; DUBÉ, M. G. **Multi-scale object-based image analysis and feature selection of multi-sensor earth observation imagery using random forests**. *International Journal of Remote Sensing*, 33(14):4502–4526, 2012.

[24] FOODY, G. M. **Status of land cover classification accuracy assessment**. *Remote Sensing of Environment*, 80(1):185–201, Apr. 2002.

[25] GEORGANOS, S.; GRIPPA, T.; VANHUYSSE, S.; LENNERT, M.; SHIMONI, M.; KALOGIROU, S.; WOLFF, E. **Less is more: Optimizing classification performance through feature selection in a very-high-resolution remote sensing object-based urban application**. *GIScience & remote sensing*, 55(2):221–242, 2018.

[26] GIESEKE, F.; ROSCA, S.; HENRIKSEN, T.; VERBESSELT, J.; OANCEA, C. E. **Massively-parallel change detection for satellite time series data with missing values**. In: *2020 IEEE 36th International Conference on Data Engineering (ICDE)*, p. 385–396. IEEE, 2020.

[27] GISLASON, P. O.; BENEDIKTSSON, J. A.; SVEINSSON, J. R. **Random forests for land cover classification**. *Pattern Recognition Letters*, 27(4):294–300, 2006.

[28] GUAN, X.; HUANG, C.; LIU, G.; MENG, X.; LIU, Q. **Mapping rice cropping systems in vietnam using an ndvi-based time-series similarity measurement based on dtw distance**. *Remote Sensing*, 8(1):19, 2016.

[29] HANSEN, M. C.; LOVELAND, T. R. **A review of large area monitoring of land cover change using landsat data**. *Remote Sensing of Environment*, 122:66–74, July 2012.

[30] HOUBORG, R.; MCCABE, M. F. **A cubesat enabled spatio-temporal enhancement method (CESTEM) utilizing planet, landsat and MODIS data**. *Remote Sensing of Environment*, 209:211–226, May 2018.

[31] JEONG, Y.-S.; JEONG, M. K.; OMITAOMU, O. A. **Weighted dynamic time warping for time series classification**. *Pattern recognition*, 44(9):2231–2240, 2011.

[32] JUSTICE, C.; TOWNSHEND, J.; VERMOTE, E.; MASUOKA, E.; WOLFE, R.; SALEOUS, N.; ROY, D.; MORISETTE, J. **An overview of modis land data processing and product status**. *Remote sensing of Environment*, 83(1-2):3–15, 2002.

[33] KARLIN, S. **The structure of dynamic programing models**. *Naval Research Logistics Quarterly*, 2(4):285–294, 1955.

[34] KATES, R. W. **What kind of a science is sustainability science?** *Proceedings of the National Academy of Sciences*, 108(49):19449–19450, 2011.

[35] LALIBERTE, A. S.; BROWNING, D.; RANGO, A. **A comparison of three feature selection methods for object-based classification of sub-decimeter resolution ultracam-l imagery**. *International Journal of Applied Earth Observation and Geoinformation*, 15:70–78, 2012.

[36] LIU, P. **A survey of remote-sensing big data**. *frontiers in Environmental Science*, 3:45, 2015.

[37] MA, L.; FU, T.; BLASCHKE, T.; LI, M.; TIEDE, D.; ZHOU, Z.; MA, X.; CHEN, D. **Evaluation of feature selection methods for object-based land cover mapping of unmanned aerial vehicle imagery using random forest and support vector machine classifiers**. *ISPRS International Journal of Geo-Information*, 6(2):51, 2017.

[38] MANABE, V. D.; MELO, M. R.; ROCHA, J. V. **Framework for mapping integrated crop-livestock systems in mato grosso, brazil**. *Remote Sensing*, 10(9):1322, 2018.

[39] MAUS, V.; CÂMARA, G.; CARTAXO, R.; SANCHEZ, A.; RAMOS, F. M.; DE QUEIROZ, G. R. **A time-weighted dynamic time warping method for land-use and land-cover mapping**. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 9(8):3729–3739, 2016.

[40] MAUS, V. **Land Use and Land Cover Monitoring Using Remote Sensing Image Time Series**. PhD thesis, PhD thesis, Instituto Nacional de Pesquisas Espaciais, Sao José dos Campos, 2016.

[41] MOUNTRAKIS, G.; IM, J.; OGOLE, C. **Support vector machines in remote sensing: A review**. *ISPRS Journal of Photogrammetry and Remote Sensing*, 66(3):247–259, 2011.

[42] NEPSTAD, D.; MCGRATH, D.; STICKLER, C.; ALENCAR, A.; AZEVEDO, A.; SWETTE, B.; BEZERRA, T.; DIGIANO, M.; SHIMADA, J.; DA MOTTA, R. S.; OTHERS. **Slowing**

amazon deforestation through public policy and interventions in beef and soy supply chains. *science*, 344(6188):1118–1123, 2014.

[43] NVIDIA. **CUDA C++ Programming Guide**. http://docs.nvidia.com/cuda/cuda-c-programming-guide/, 2020.

[44] NVIDIA, C. **Cuda c programming guide, version 9.1**. *NVIDIA Corp*, 2018.

[45] OLIVEIRA, S. S. T. D.; OTHERS. **Explorando paralelismo em big data no processamento de séries temporais de imagens de sensoriamento remoto**. 2019.

[46] OLIVEIRA, S. S.; CARDOSO, M. D. C.; BUENO, E.; RODRIGUES, V. J.; MARTINS, W. S. **Exploiting parallelism to generate meta-features for land use and land cover classification with remote sensing time series**. *Brazilian Symposium on Geoinformatics (GeoInfo)*, p. 135–146, 2019.

[47] OLIVEIRA, S. S.; PASCOAL, L. M. L.; FERREIRA, L.; DE CASTRO CARDOSO, M.; BUENO, E. F.; VAGNER, J.; MARTINS, W. S. **Sp-twdtw: A new parallel algorithm for spatio-temporal analysis of remote sensing images.** In: *GEOINFO*, p. 46–57, 2018.

[48] OLIVEIRA, S. S.; PASCOAL, L. M.; FERREIRA, L.; CARDOSO, M. D. C.; BUENO, E.; RODRIGUES, V. J.; MARTINS, W. S. **Sp-twdtw: A new parallel algorithm for spatio-temporal analysis of remote sensing images**. *Brazilian Symposium on Geoinformatics (GeoInfo)*, p. 46–57, 2018.

[49] PAIVA, R.; OLIVEIRA, S.; MARTINS, W.; PARENTE, L. **Análise de metacaracterísticas para classificação de uso e cobertura do solo utilizando random forest**. In: *Anais do XI Workshop de Computação Aplicada à Gestão do Meio Ambiente e Recursos Naturais*, p. 71–80. SBC, 2020.

[50] PAL, M. **Random forest classifier for remote sensing classification**. *International Journal of Remote Sensing*, 26(1):217–222, 2005.

[51] PARENTE, L.; FERREIRA, L. **Assessing the spatial and occupation dynamics of the brazilian pasturelands based on the automated classification of modis images from 2000 to 2016**. *Remote Sensing*, 10(4):606, 2018.

[52] PARENTE, L.; MESQUITA, V.; MIZIARA, F.; BAUMANN, L.; FERREIRA, L. **Assessing the pasturelands and livestock dynamics in brazil, from 1985 to 2017: A novel approach based on high spatial resolution imagery and google earth engine cloud computing**. *Remote Sensing of Environment*, 232:111301, 2019.

[53] PEDREGOSA, F.; VAROQUAUX, G.; GRAMFORT, A.; MICHEL, V.; THIRION, B.; GRISEL, O.; BLONDEL, M.; PRETTENHOFER, P.; WEISS, R.; DUBOURG, V.; VANDERPLAS, J.; PASSOS, A.; COURNAPEAU, D.; BRUCHER, M.; PERROT, M.; DUCHESNAY, E. **Scikit-learn: Machine learning in Python**. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[54] PETITJEAN, F.; FORESTIER, G.; WEBB, G. I.; NICHOLSON, A. E.; CHEN, Y.; KEOGH, E. **Dynamic time warping averaging of time series allows faster and more accurate classification**. In: *2014 IEEE international conference on data mining*, p. 470–479. IEEE, 2014.

[55] PFISTER, G. F. **In search of clusters: the coming battle in lowly parallel computing**. Prentice-Hall, Inc., 1995.

[56] PICOLI, M. C. A.; CAMARA, G.; SANCHES, I.; SIMÕES, R.; CARVALHO, A.; MACIEL, A.; COUTINHO, A.; ESQUERDO, J.; ANTUNES, J.; BEGOTTI, R. A.; OTHERS. **Big earth observation time series analysis for monitoring brazilian agriculture**. *ISPRS journal of photogrammetry and remote sensing*, 145:328–339, 2018.

[57] PLAZA, A. **High performance computing in remote sensing**. Chapman & Hall/CRC, Boca Raton, FL, 2008.

[58] PRZYMUS, P.; KACZMARSKI, K. **Dynamic compression strategy for time series database using gpu**. In: *New Trends in Databases and Information Systems*, p. 235–244. Springer, 2014.

[59] RABINER, L.; ROSENBERG, A.; LEVINSON, S. **Considerations in dynamic time warping algorithms for discrete word recognition**. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 26(6):575–582, 1978.

[60] RODRIGUEZ-GALIANO, V. F.; GHIMIRE, B.; ROGAN, J.; CHICA-OLMO, M.; RIGOL-SANCHEZ, J. P. **An assessment of the effectiveness of a random forest classifier for land-cover classification**. *ISPRS Journal of Photogrammetry and Remote Sensing*, 67:93–104, 2012.

[61] ROMANI, L. A.; GONCALVES, R.; ZULLO, J.; TRAINA, C.; TRAINA, A. J. **New dtw-based method to similarity search in sugar cane regions represented by climate and remote sensing time series**. In: *2010 IEEE International Geoscience and Remote Sensing Symposium*, p. 355–358. IEEE, 2010.

[62] SAKOE, H.; CHIBA, S. **Dynamic programming algorithm optimization for spoken word recognition**. *IEEE transactions on acoustics, speech, and signal processing*, 26(1):43–49, 1978.

[63] SANDES, E. F. D. O. **Algoritmos paralelos exatos e otimizações para alinhamento de sequências biológicas longas em plataformas de alto desempenho**. 2015.

[64] SHUMWAY, R. H.; STOFFER, D. S. **Time series and its applications: With r examples**. *Springer Science*, p. 1–202, 2011.

[65] TSAI, Y. H.; STOW, D.; CHEN, H. L.; LEWISON, R.; AN, L.; SHI, L. **Mapping vegetation and land use types in fanjingshan national nature reserve using google earth engine**. *Remote Sensing*, 10(6):927, 2018.

[66] VAPNIK, V. N. **The Nature of Statistical Learning Theory**. Springer New York, 1995.

[67] WEGNER MAUS, V.; CÂMARA, G.; APPEL, M.; PEBESMA, E. **dtwsat: Time-weighted dynamic time warping for satellite image time series analysis in r**. *Journal of Statistical Software*, 88(5):1–31, 2019.

[68] WIENS, T. S.; DALE, B. C.; BOYCE, M. S.; KERSHAW, G. P. **Three way k-fold cross-validation of resource selection functions**. *Ecological Modelling*, 212(3-4):244–255, 2008.