Unioeste - Universidade Estadual do Oeste do Paraná
CENTRO DE CIÊNCIAS EXATAS E TECNOLÓGICAS
Colegiado de Ciência da Computação
Curso de Bacharelado em Ciência da Computação

Proposta de implementação multithread e com redução de uso de memória do teste de Knox para clusters espaço-temporais

Juliano Felipe Prass da Silva

CASCAVEL
2018

JULIANO FELIPE PRASS DA SILVA

PROPOSTA DE IMPLEMENTAÇÃO MULTITHREAD E COM REDUÇÃO DO USO DE MEMÓRIA DO TESTE DE KNOX PARA CLUSTERS ESPAÇO-TEMPORAIS

Monografia apresentada como requisito parcial para obtenção do grau de Bacharel em Ciência da Computação, do Centro de Ciências Exatas e Tecnológicas da Universidade Estadual do Oeste do Paraná - Campus de Cascavel

Orientador: Prof. Marcio Seiji Oyamada

CASCAVEL
2018

JULIANO FELIPE PRASS DA SILVA

PROPOSTA DE IMPLEMENTAÇÃO MULTITHREAD E COM REDUÇÃO DO USO DE MEMÓRIA DO TESTE DE KNOX PARA CLUSTERS ESPAÇO-TEMPORAIS

Monografia apresentada como requisito parcial para obtenção do Título de Bacharel em Ciência da Computação, pela Universidade Estadual do Oeste do Paraná, Campus de Cascavel, aprovada pela Comissão formada pelos professores:

Prof. Marcio Seiji Oyamada (Orientador)
Colegiado de Ciência da Computação,
UNIOESTE

Prof. Rogério Luis Rizzi
Colegiado de Matemática, UNIOESTE

Prof. Guilherme Galante
Colegiado de Ciência da Computação,
UNIOESTE

Cascavel, 14 de dezembro de 2018

DEDICATÓRIA

Dedico ao meu pai, por sempre ter algo para me ensinar

Why do we fall, sir? So that we can learn that with great power comes great responsibility

Lista de Figuras

3.1 Fluxograma geral do algoritmo de Knox. 24
3.2 Fluxograma do cálculo das adjacências 25
3.3 Fluxograma do cálculo de n_{st} 27
3.4 Cálculo dos termos n_{2s} e n_{2t} com complexidade quadrática. 28
3.5 Matriz de adjacência de exemplo 29
3.6 Iterações sob as posições i, j da matriz de exemplo. 30
3.7 Exemplo para a linha $i = 0$. 32
3.8 Exemplo de matrizes de adjacência completa e parcial e contadores de n_2 . . . 33
3.9 Bloco mínimo de exemplo sem contadores 34
3.10 Bloco mínimo de exemplo sem contadores em matriz parcial 34
3.11 Fluxograma para amostragem Fisher-Yates 36
3.12 Esquema de particionamento para amostragem 37
3.13 Matriz de exemplo para a presente seção. 37
3.14 Bloco para processamento e bloco para cálculo dos termos n_2 sem contadores . 38
3.15 Segunda iteração do algoritmo em blocos. 39
3.16 Excemplo de redundância de processamento em matrizes completas 39
3.17 Meia matriz de exemplo. 40
3.18 Primeira iteração no algoritmo com meia matriz. 40
3.19 Segunda iteração no algoritmo com meia matriz. 41
3.20 Fluxograma para o algoritmo de blocos com contadores e matrizes parciais . . 42
3.21 Fluxograma geral do iKnox 43

4.1 Distribuição espacial dos casos de Cascavel. 56
4.2 Gráfico da distribuição da estatística de iKnox para Cascavel 57

4.3 Gráfico da distribuição da estatística de Z para Cascavel 60
4.4 Distribuição espacial dos casos do Rio de Janeiro. 65
4.5 Gráfico da distribuição da estatística de iKnox para Rio de Janeiro 65
4.6 Gráfico de distribuição dos valores de Z para Rio de Janeiro 66
4.7 Gráfico de pico de memória das implementações *single thread* testadas. 73
4.8 Gráfico de pico de memória das implementações *multithread* testadas. 73
4.9 Gráfico do tempo de execução das implementações *single thread* testadas. . . . 74
4.10 Gráfico do tempo de execução das implementações *multithread* testadas. 74
4.11 Gráfico de pico de memória da implementação de Höhle et al. (2017) 75
4.12 Gráfico de tempo de execução da implementação de Höhle et al. (2017), *single vs. multithread* 75
4.13 Gráfico de pico de memória da implementação proposta, *single vs. multithread*. 76

B.1 Modelo dos fluxogramas. 84

Lista de Tabelas

2.1 Esquema de tabela de contingência 6
2.2 Exemplo recortado de Boza (2009). 12
2.3 Distâncias entre os casos de exemplo 12
2.4 Adjacências de todas as duplas por extenso. 13
2.5 Adjacências em termos de a_{ij}^S e a_{ij}^T 13
2.6 Tabela de contingência do exemplo. 14
2.7 Simulações de MC do exemplo 15
2.8 Contagem do *rank* das simulações de MC do exemplo 16
2.9 Resultados obtidos no exemplo. 16
2.10 Limiares críticos para testes do iKnox. 19
2.11 Resultados dos testes de iKnox de exemplo. 20
2.12 Risco Elevado do exemplo do iKnox 20

3.1 Matriz completa com adjacências espaciais de exemplo 45
3.2 Matriz completa com adjacências temporais de exemplo 45
3.3 Vetores com adjacências espaciais e temporais. 45
3.4 Matriz parcial com adjacências espaciais de exemplo 46
3.5 Matriz parcial com adjacências espaciais de exemplo 46
3.6 Matriz com adjacências espaciais e um contador. 47
3.7 Matriz com adjacências espaciais e dois contadores. 47
3.8 Matriz parcial com adjacências temporais amostrada. 48
3.9 Divisão da matriz de adjacência de exemplo em blocos 49
3.10 Divisão do vetor de adjacências de exemplo em blocos 49
3.11 Blocos e contadores da primeira iteração do exemplo 51

3.12 Contadores de n_2 após primeira iteração do exemplo 51
3.13 Contadores de MC após primeira iteração do exemplo 52
3.14 Blocos e contadores da segunda iteração do exemplo 52
3.15 Contadores de n_2 após segunda iteração do exemplo 52
3.16 Blocos de vetores e contadores da primeira iteração do exemplo 53

4.1 Primeira parte das estatísticas de iKnox para Cascavel 58
4.2 Segunda parte das estatísticas de iKnox para Cascavel 59
4.3 Primeira parte dos valorese de Z para Cascavel 60
4.4 Segunda parte dos valores de Z para Cascavel 61
4.5 Primeira parte dos valores de $p - Valor_{MC}$ para Cascavel 62
4.6 Segunda parte dos valores de $p - Valor_{MC}$ para Cascavel 63
4.7 Parte 1 da tabela de ER de Cascavel. 63
4.8 Parte 2 da tabela de ER de Cascavel. 64
4.9 Parte 3 da tabela de ER de Cascavel. 64
4.10 Legenda das tabelas ER de Cascavel 64
4.11 Valores da estatística de iKnox para Rio de Janeiro 66
4.12 Valores de Z para Rio de Janeiro 67
4.13 Valores do $p - Valor_{MC}$ para Rio de Janeiro 68
4.14 Resumo das otimizações da solução proposta. 68
4.15 Pico de memória e tempo de execução para implementação de Tango (2010) . . 70
4.16 Pico de memória para implementação de Höhle et al. (2017) e a proposta 71
4.17 Tempo de execução para implementação de Höhle et al. (2017) e a proposta . . 72

A.1 Primeira parte das amostragens de MC para o exemplo 79
A.2 Segunda parte das amostragens de MC para o exemplo 80
A.3 Adjacências espaciais do exemplo de iKnox 80
A.4 Primeira parte das adjacências temporais e suas amostragens do exemplo de iKnox 81
A.5 Segunda parte das adjacências temporais e suas amostragens do exemplo de iKnox 81
A.6 Terceira parte das adjacências temporais e suas amostragens do exemplo de iKnox 82

A.7 Resumo das adjacências espaço-temporais observadas do exemplo de iKnox, onde as letras na base são usadas para referenciar cada vetor de adjacência na tabela ER posterior. 82
A.8 Relação das tabelas de adjacência em uma tabela ER 83

C.1 Primeiro bloco de exemplo para o componente 86
C.2 Primeira amostragem do bloco de exemplo 86
C.3 Último bloco de exemplo para o componente 87

Lista de Abreviaturas e Siglas

CPU	*Central Processing Unit*
ER	*Excess Risk*
FPGA	*Field Programmable Gate Array*
GPU	*Graphics Processing Unit*
iKnox	*Incremental Knox Test*
LUT	*Look-up-table*
MC	Monte Carlo
Nrep	Número de Repetições
UTM	*Universal Transversa de Mercator*

Lista de Símbolos

n	Número de casos de entrada do teste de Knox
N	Número de maneiras em que os n casos podem ser juntados em pares
n_s	Pares próximos no espaço
δ_S	Limiar crítico espacial
n_t	Pares próximos no tempo
δ_T	Limiar crítico temporal
n_{st}	Estatística de Knox, pares próximos no espaço e no tempo
d_{ij}^S	Distância espacial entre os casos i e j
d_{ij}^T	Distância temporal entre os casos i e j
a_{ij}^S	Proximidade espacial dos casos i e j
a_{ij}^T	Proximidade temporal dos casos i e j
$E(n_{st})$	Esperança matemática do teste de Knox
$Var(n_{st})$	Variância do teste de Knox
$\xi(n_{st})$	Primeiro momento do teste de Knox
$\xi(n_{st}^2))$	Segundo momento do teste de Knox
n_{2s}	Número de vezes que dois pares de casos próximo no espaço são contíguos
n_{2t}	Número de vezes que dois pares de casos próximo no tempo são contíguos
$p - Valor_{Poisson}$	Valor do pValor calculado por distribuição de Poisson
λ	Equivalente à $E(n_{st})$
R	*rank* de n_{st} para o conjunto resultante do procedimento de MC
$p - Valor_{MC}$	Valor do pValor calculado por simulações de MC
n_{st_v}	v-gésima estatística simulada de MC
b_{ij}^T	Proximidade temporal de iKnox dos casos i e j
IK_{st}	Estatística de Knox para a variação *iKnox*
μ_{MC}	Média dos n_{st} obtidos nas simulações de MC
Z_{IK}	Estatística de iKnox normalizada
SE_{MC}	Erro padrão do conjunto utilizado para calcular μ_{mc}
SD_{MC}	Desvio padrão do conjunto utilizado para calcular μ_{mc}
n_2	Generalização para se referir à n_{2s} e n_{2t} simultaneamente
S	Vetor de adjacências espaciais
T	Vetor de adjacências temporais
ST	Vetor de adjacências espaço-temporais

Sumário

Lista de Figuras vi

Lista de Tabelas viii

Lista de Abreviaturas e Siglas xi

Lista de Símbolos xii

Sumário xiii

Resumo xv

1 Introdução 1

1.1 Objetivos 2

1.2 Organização do texto 2

2 Clusterização Espaço-Temporal 3

2.1 Método de Knox 4

2.1.1 P-Valor e Simulações de Monte Carlo 8

2.1.2 Exemplo 12

2.2 Teste de Knox Incremental 16

2.2.1 Exemplo 18

3 Implementações e otimizações sobre o Teste de Knox 21

3.1 Algoritmo e Fluxograma Geral 23

3.2 Cálculo da Média e da Variância da Aproximação de Poisson 28

3.3 Amostragem ao Cálculo das Simulações de Monte Carlo 35

3.4 Estratégia em Blocos 37

3.5 Algoritmo Incremental 42

3.6 Exemplo 44

3.6.1 Estatística de Knox 45

3.6.2 Variância 46
3.6.3 Simulação de Monte Carlo 48
3.6.4 Organização em blocos 48
4 Resultados **54**
4.1 Clusterização espaço-temporal de Dengue 54
4.1.1 Caso Cascavel 56
4.1.2 Caso Rio de Janeiro 64
4.2 Aspectos Computacionais 68
5 Conclusões **77**
A Simulações de Monte Carlo utilizadas nas seções de exemplo **79**
B Modelo dos fluxogramas utilizados **84**
C Arquitetura para FPGAs **85**
C.1 Exemplo 86
Referências Bibliográficas **88**

Resumo

O teste de Knox é utilizado para a clusterização espaço-temporal de dados, no entanto, as soluções existentes não apresentam escalabilidade para memória utilizada e tempo de processamento com o aumento do número de casos entrada. A solução aqui proposta apresenta melhorias nestes fatores em relação à implementação de Meyer (2017). No uso de memória, o ganho se dá pela utilização de tipos de dados com menor espaço em memória e divisão de dados em blocos. No tempo de execução, o ganho se dá como consequência da organização e representação dos dados e no paralelismo de dados extraído da aplicação. Em um teste com entrada n = 10.000, 4 *threads* e 999 simulações de Monte Carlo, a redução do uso de memória foi aproximadamente 800% e de tempo de execução foi cerca de 54%.

Palavras-chave: Clusterização Espaço-Temporal; Teste de Knox; Paralelismo de dados.

Capítulo 1

Introdução

A clusterização é utilizada para testar a hipótese da existência de relação entre as posições de casos de entrada em uma ou mais dimensões, como espaço e tempo, no caso da clusterização espaço-temporal. Dentre diferentes testes dessa categoria, o teste de Knox é utilizado desde meados da década de 60 devido a sua relativa simplicidade, uma vez que apenas necessita das posições no espaço e no tempo dos casos a serem estudados e de limiares pré-estabelecidos que indicam a faixa de distâncias que são consideradas próximas ou não. Tal simplicidade, no entanto, padece de limitações. Do ponto de vista do método em si, há críticas quanto a arbitrariedade na escolha dos limiares e da ineficiência do método, e quando observa-se implementações existentes, surgem problemas com testes em que há grande quantia de casos de entrada.

O teste realiza a comparação exaustiva entre todas as combinações possíveis entre dois casos, dados n entradas, para cada dimensão considerada, isto é, as distâncias no espaço e no tempo para as N duplas possíveis. A complexidade dessa etapa é, então quadrática, visto que há como combinar n casos de $N = (n \cdot (n-1))/2$ maneiras. Assim, algoritmos que não realizam nenhum tipo de otimização em relação a complexidade espacial ou temporal, podem não resolver em tempo razoável quando para entradas com n grande. Na bibliografia, os primeiros usos do método possuem um n relativamente pequeno, a exemplo do estudo por Knox (1964) e David e Barton (1966) com apenas $n = 96$, totalizando 4560 cálculos de distâncias. No entanto, aplicando-se o método com 200.000 casos ter-se-iam 19.999.900.000 cálculos de distância, aumentando consideravelmente o tempo de execução e consumo de memória.

Como as distâncias temporal e espacial entre dois casos não tem nenhuma relação/interferência com a distâncias entre quaisquer outros dois casos, o método pode ser classificado como tendo paralelismo de dados, não de tarefas. Consequentemente, o método de

Knox é, pelo menos no cálculo das distâncias, facilmente paralelizável, indicando, também, que é viável a aplicação de técnicas de particionamento nos dados presentes em memória para serem processados imediatamente, reduzindo o consumo de memória e possibilitando que casos maiores sejam executados em tempo hábil.

1.1 Objetivos

Este trabalho tem como objetivo desenvolver uma implementação do teste de Knox que apresente melhor desempenho em relação à complexidade espacial e temporal visando a redução do uso de memória e melhorias de desempenho na execução *multithread*.

1.2 Organização do texto

O Capítulo 2 apresenta, de maneira geral, o método de Knox. O Capítulo 3 apresenta as implementações do método de Knox e considerações sobre otimizações em relação à códigos presentes na literatura. O Capítulo 4 apresenta dois estudos de caso utilizando a solução proposta, testes das implementações existentes e da proposta e conclusões acerca do desempenho das mesmas. Por fim, o Capítulo 5 apresenta as conclusões e trabalhos futuros.

Capítulo 2

Clusterização Espaço-Temporal

Em essência, a clusterização é um processo realizado para testar a hipótese da existência ou não de relação entre as posições dos casos de entrada em uma ou mais dimensões, seja ela espaço, tempo ou ambos. Testar somente em uma dimensão, assume, por sua vez, que as outras dimensões consideradas são fixadas o que pode ter resultados limitados para estudos onde não há um comportamento estável para a(s) dimesão(ões) fixadas. De qualquer forma, a aplicação de um teste do tipo pode determinar se os pontos estão distribuídos aleatoriamente pela(s) dimensão(ões) consideradas ou se há uma *clusterização* entre os mesmos. Com essas considerações, pode existir um caso em que aparentemente há uma relação entre os casos próximos, mas que um teste de clusterização pode definir como distribuição aleatória, ou o contrário, em que aparentemente não há relação mas o teste aponta uma não aleatoriedade, tudo dependendo dos parâmetros, muitas vezes arbitrários, que determinam o que é próximo ou não.

Para determinar a existência ou não nas relações entre os casos, considera-se uma tese matemática que para ser provada pode-se utilizar o método de demonstração por absurdo, em que se mostra que a falsidade da conclusão leva a uma contradição lógica ou que a falsidade da hipótese, que deve ser sempre verdadeira. De forma análoga também pode-se testar uma hipótese estatística pressupondo o oposto do que se quer testar, a existência de interação de casos de teste em uma ou mais dimensões, ou seja, supõe-se que não há interação alguma entre os casos. Se uma contradição for encontrada, então conclui-se que o pressuposto está errado, e a hipótese inicial, que há interação, não pode ser descartada. Assim determina-se o teste de hipótese que é inerente aos testes de clusterização.

De um ponto de vista mais exato, a clusterização espaço-temporal é usada para verificar a existência de interação dos casos tanto na dimensão do espaço quanto na do tempo, de modo

que a hipótese inicial e a hipótese alternativa são como:

1. H_0: Não há interação espaço-temporal;

2. H_a: Há interação espaço-temporal.

2.1 Método de Knox

O teste mais frequentemente empregado para clusterização espaço-temporal é o de Knox proposto por Knox e Bartlett (1964), em que o teste foi utilizado para estudar a clusterização de leucemia, como em Knox (1964). Knox conjecturou que a estatística do teste seguia uma distribuição de Poisson para as hipóteses inicial e alternativa, o que foi provado por David e Barton (1966), que também propuseram outro teste de clusterização espaço-temporal.

Outros autores como Mantel (1967) e Pike e Smith (1968) realizaram generalizações do teste de Knox. O primeiro assume que o risco diminui conforme a distância aumenta, e o segundo considera períodos latentes infecciosos e possíveis movimentos no espaço durante o período em estudo. Tais testes possuem parâmetros desconhecidos que são utilizados para determinar a proximidade dos casos requerendo, consequentemente, múltiplos testes à escolha de valores plausíveis (TANGO, 2010).

Assim como discutido por Silva (2011), especificando-se as distâncias críticas espacial e temporal, é viável determinar se um par de eventos está próximo espaço-temporalmente, visto que o teste se baseia na quantidade de pares de eventos que estão simultaneamente próximos no tempo e no espaço. Um alto valor desses pares indica que há uma tendência de casos próximos no tempo estarem também próximos no espaço.

Contudo, como apontado por Silva (2011), o teste de Knox, assim como outras técnicas para testar a hipótese de independência entre espaço e tempo, também padece do problema típico dos testes de hipóteses para o caso de existirem muitos eventos, que é a indicação que o teste é significativo, mesmo se a associação espaço-temporal for fraca.

O teste de Knox é fundamentado na contagem da quantidade de pares de eventos que ocorrem em intervalos de tempo e distância críticas pré-especificados considerando-se n pontos e, então, existem N distintos pares de pontos, calculados pela equação (2.1). Para formalizar tal processo, considera-se que n_s seja a quantidade observada de pares de eventos que estão próxi-

mos no espaço, que são aqueles pares de pontos separados por uma distância espacial menor que a distância espacial crítica, δ_S. E n_t a quantidade observada de pares de eventos que estão próximos no tempo que são aqueles pares de pontos separados por uma distância temporal menor do que a distância temporal crítica, δ_T. Tais distâncias críticas δ_S e δ_T devem ser especificadas pelo usuário de acordo com o conhecimento do processo, e são cruciais à adequada análise da ocorrência ou não da interação espaço-temporal.

$$N = \frac{n(n-1)}{2} \tag{2.1}$$

A estatística do teste de Knox é simplesmente o valor n_{st}, que é a quantidade observada de pares de eventos que são simultaneamente próximos no espaço e no tempo. A estatística de teste excede seu valor esperado quando pontos que estão próximos no espaço estão mais próximos no tempo que o esperado, como mostrado por Silva (2011). Esta estatística é comparada com uma distribuição de referência, sob a hipótese nula de que o processo não apresenta interação espaço-temporal, obtida por meio de permutações aleatórias dos índices dos eventos originais por meio de um processo tipo Monte Carlo. Um valor pequeno para $p - Valor$ é uma evidência a favor da hipótese de ocorrer interação espaço-temporal.

O método de clusterização espaço-temporal de Knox tem a vantagem de que não é necessário conhecer o tamanho nem as características da população em estudo. Para desenvolver e aplicar o referido método é necessário:

1. Considerar n casos, sendo que para cada deles deve-se conhecer suas coordenadas espaciais $\{(x_1, y_1), \cdots, (x_n, y_n)\}$ correspondentes a cada um dos casos do evento ocorrido durante o período de estudo. Além disso, deve-se conhecer as datas $\{t_1, \cdots, t_m\}$ (dd/mm/aaaa) de ocorrência desses casos.

2. Determinar a distância no espaço, d_{ij}^S, e no tempo, d_{ij}^T para cada par de pontos e informar os limiares críticos espacial, δ_S, e temporal, δ_T.

3. Classificar os $N = n(n-1)/2$ pares de acordo com o seguinte critério:

   - Um par de casos está próximo no espaço e no tempo se $d_{ij}^S \leqslant \delta_S$ e $d_{ij}^T \leqslant \delta_T$.
   - Um par de casos está próximo no tempo se $d_{ij}^T \leqslant \delta_T$.

- Um par de casos está próximo no espaço se $d_{ij}^S \leqslant \delta_S$.
- Um par de casos não está próximo no espaço nem no tempo se $d_{ij}^S > \delta_S$ e $d_{ij}^T > \delta_T$.

4. Construir a Tabela de contingência, como a Tabela (2.1), em que:
   - n é a quantidade total de casos observados do evento ocorrido.
   - n_{st} é a quantidade observada de pares de casos que são simultaneamente próximos no espaço e no tempo.
   - n_s é a quantidade observada de pares de casos que estão próximos no espaço.
   - n_t é a quantidade observada de pares de casos que estão próximos no tempo.
   - $n - n_s - n_t + n_{st}$ é a quantidade observada de pares de casos que não estão próximos no espaço nem no tempo, obtida diretamente a partir do conhecimento das quantidades n_s, n_t e n_{st}.
5. O teste estatístico é a quantidade observada de pares de casos que estão simultaneamente próximos no espaço e no tempo, n_{st}, sendo as distribuições do qui-quadrado, de Poisson, e da aproximação da distribuição Normal à distribuição de Poisson as mais usualmente empregados no método de Knox.

Tabela 2.1: Tabela de contingência 2×2, à disposição dos N pares segundo as quatro possibilidades de proximidades espacial e temporal.

Temporal \ Espacial	Próximos	Não próximos	Total
Próximos	n_{st}	$n_t - n_{st}$	n_t
Não próximos	$n_s - n_{st}$	$N - n_s - n_t + n_{st}$	$N - n_t$
Total	n_s	$N - n_s$	N

Algoritmicamente, para expressar a estatística do teste de Knox, que é simplesmente o valor n_{st} dos pares observados que estão próximos do espaço e do tempo, é conveniente definir métricas à proximidade espacial a_{ij}^S e proximidade temporal a_{ij}^T, como em (2.2) e (2.3), apresentadas em (TANGO, 2010).

$$a_{ij}^S = \begin{cases} 1, i \neq j \; d_{ij}^S < \delta_S \\ 0, \text{caso contrário} \end{cases} \tag{2.2}$$

$$a_{ij}^{T} = \begin{cases} 1, i \neq j \text{ e } d_{ij}^{T} < \delta_{T} \\ 0, \text{caso contrário} \end{cases} \tag{2.3}$$

A estatística do teste de Knox, denotada por n_{st}, é determinada nessa formulação algorítmica como na expressão (2.4) (TANGO, 2010).

$$n_{st} = \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} a_{ij}^{S} a_{ij}^{T} \tag{2.4}$$

Para realizar determinadas avaliações é necessário calcular, além da estatística de Knox, n_{st}, o seu Valor Esperado (Esperança Matemática), $E(n_{st})$, e sua Variância, $Var(n_{st})$. Para tal são utilizadas expressões para o primeiro momento, $\xi(n_{st})$, e segundo momento, $\xi(n_{st}^2)$, derivados por David e Barton (1966). Tais expressões são, respectivamente, (2.5) e (2.6).

$$\xi(n_{st}) = \frac{2n_s n_t}{n(n-1)} \tag{2.5}$$

e

$$\xi(n_{st}^2) = \frac{2n_s n_t}{n(n-1)} + \frac{4n_{2s} n_{2t}}{n(n-1)(n-2)} + \frac{4\left[n_s(n_s-1) - 2n_{2s}\right]\left[n_t(n_t-1) - 2n_{2t}\right]}{n(n-1)(n-2)(n-3)} \tag{2.6}$$

Pode-se notar, no entanto, que Tango (2010) apresenta diferentemente a formulação para o segundo momento, como visto na equação (2.7). Delas, utiliza-se a primeira.

$$\xi(n_{st}^2) = \frac{2n_s n_t}{n(n-1)} + \frac{4n_{2s} n_{2t}}{n(n-1)(n-2)} + \frac{4\left[n_s(n_s-1) - n_{2s}\right]\left[n_t(n_t-1) - n_{2t}\right]}{n(n-1)(n-2)(n-3)} \tag{2.7}$$

Os termos n_s, n_t, n_{2s} e n_{2t}, utilizados em tais expressões podem ser calculadas como apresentadas em Tango (2010), por (2.8)-(2.11), em que n_{2s} denota o número de vezes que dois pares de casos próximos no espaço são contíguos e n_{2t} denota o equivalente para o tempo.

$$n_s = \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} a_{ij}^{S} \tag{2.8}$$

$$n_t = \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} a_{ij}^{T} \tag{2.9}$$

$$n_{2s} = \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{k \neq j}^{n} a_{ij}^{S} a_{ik}^{S} \tag{2.10}$$

$$n_{2t} = \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{k \neq j}^{n} a_{ij}^{T} a_{ik}^{T} \tag{2.11}$$

Assim sendo, o método de Knox, as hipóteses formuladas que devem ser confrontadas aos dados, segundo os testes estatísticos, são:

- H_0: Os dados seguem uma distribuição de Poisson, e portanto não ocorre a clusterização espaço-temporal.
- H_a: Os dados não seguem uma distribuição de Poisson, ocorrendo a clusterização espaço-temporal.

2.1.1 P-Valor e Simulações de Monte Carlo

Dado qualquer Teste de Hipótese, torna-se necessário, a especificação da distribuição da Estatística usada. Alguns testes são baseados na distribuição nula exata, distribuição de T ou distribuição de F, derivados sob as assunções dos dados utilizados. Distribuição nula é a distribuição amostral da estatística do teste sob a hipótese H_0. As distribuições nulas são derivadas sob argumentos assintóticos, mas para certos casos as assunções empregadas para tal podem ser inapropriadas ou não acuradas. Pode até mesmo ser difícil uma derivação da suposta distribuição.

Para estes casos, pode-se empregar um Teste de Hipóteses baseado em simulações de Monte Carlo (TANGO, 2010). No caso específico do método de Knox, o emprego de tal abordagem é realizado considerando-se o trabalho de David e Barton (1966), que mostraram que a estatística n_{st} segue uma distribuição de Poisson para amostras grandes, tendo, consequentemente, média aproximadamente igual a variância.

Portanto, como a variável tempo é tal que $t \geqslant 0$, o $p-$Valor unilateral à direita pode ser calculado aproximadamente como visto em (2.12).

$$p - Valor_{poisson} \approx Poisson(\xi(n_{st}) > n_{st}) \tag{2.12}$$

em que $\xi(n_{st})$ é o primeiro momento ou o valor esperado, usualmente denotado na distribuição

de Poisson como λ. Assim sendo, pode-se empregar uma distribuição de Poisson definida em (2.13), em que e é base do logaritmo natural.

$$f(n_{st}, \lambda) = \frac{e^{-\lambda}\lambda^{n_{st}}}{n_{st}!} \tag{2.13}$$

Assim, calcula-se o $p-$Valor como (2.14) ou, equivalentemente, como (2.15).

$$p - Valor_{Poisson} = Pr(Poisson(\lambda) \leqslant n_{st}) \tag{2.14}$$

$$p - Valor_{Poisson} = 1 - Pr(Poisson(\lambda) > n_{st}) \tag{2.15}$$

Para calcular o $p-$Valor simulado por um método de Monte Carlo, deve-se, considerar os seguintes passos (TANGO, 2010):

- Inicialmente deve-se calcular a Estatística Observada baseada nos dados de entrada.
- Após, realiza-se uma amostra aleatória em N_{rep} conjuntos de teste a partir do mesmo conjunto utilizado no primeiro passo. Calcula-se a estatística para cada um destes conjuntos, armazenando-as como as estatísticas simuladas.
- Por fim, o $p-$Valor unilateral à direita é calculado como uma média aritmética simples do conjunto que possui as estatísticas simuladas e a observada.

Assim sendo, o $p-$Valor simulado é calculado pela equação (2.16), em que o termo R é obtido pela equação (2.17) (TANGO, 2010).

$$p - Valor_{MC} = \frac{R}{N_{rep} + 1} \tag{2.16}$$

com

$$R = 1 + \sum_{k=1}^{N_{rep}} (n_{st_v} \geqslant n_{st}) \tag{2.17}$$

em que $p - Valor_{MC}$ é o $p-$Valor simulado pelo método Monte Carlo, e n_{st_v} é v-gésima Estatística Simulada, e n_{st} é a Estatística Observada.

Note-se que N_{rep} é o número de permutações de Monte Carlo, e R denota o *Rank* das permutações, que é definido como quantos valores do conjunto das estatísticas das permutações

de Monte Carlo e da Estatística Observada são maiores ou iguais à própria estatística observada. Mostra-se, numericamente, que pelo menos um valor pertencente a este conjunto é igual a estatística observada, visto que ela mesma é parte do conjunto e, consequentemente, o $p-Valor$ utilizando a metodologia de Monte Carlo nunca será igual à 0.

Ainda, na discussão sobre a não nulidade do valor do $p-Valor$ calculado por Monte Carlo, tem-se que a Estatística Observada é sempre igual a si mesma, portanto não é necessário incluí-la no conjunto de estatísticas simuladas, substituindo a comparação por um incremento de 1 no valor de R. Ademais, a divisão do R por N_{rep} também deve sofrer o mesmo incremento, uma vez que o último valor corresponde ao tamanho do conjunto utilizado para obter R, que não possui a Estatística Observada explicitamente, mas tem o número de elementos aumentado.

Como exemplo para o cálculo do valor de $p-$Valor, considera-se que em um conjunto com N_{rep} estatísticas simuladas onde todas são menores que a Observada, tem-se $R = 0$. O primeiro item não inclui a Observada, portanto não apresenta o incremento à R e N_{rep}. O segundo, no entanto, inclui, apresentando os devidos incrementos. Assim para $Nrep = 999$ tem-se:

- $p - Valor_{MC} = \dfrac{R}{N_{rep}} = \dfrac{0}{999} = 0.$
- $p - Valor_{MC} = \dfrac{R+1}{N_{rep}+1} = \dfrac{0+1}{1000} = 0,001.$

mostrando que se não incluir a n_{st} no ***Rank***, na sua divisão por N_{rep} obteria-se 0, enquanto a inclusão garantiria que o resultado fosse sempre diferente de zero, e nesse caso, igual à $0,001$.

Quando a Estatística Observada é menor ou igual à todas as Estatísticas Simuladas/Calculadas então $R = 999$ e, assim:

- $p - Valor_{MC} = \dfrac{R}{N_{rep}} = \dfrac{999}{999} = 1.$
- $p - Valor_{MC} = \dfrac{R+1}{N_{rep}+1} = \dfrac{999+1}{1000} = 1.$

mostrando que não ocorre qualquer alteração nos resultados para estes casos.

Esses resultados vêm ao encontro da argumentação de North, Curtis e Sham (2003) para defender a formulação $(r+1)/(n+1)$ como uma estimativa *não tendenciosa*, *unbiased*, para estimar o $p-$Valor quando do emprego do método de Monte Carlo, em que n é o número de simulações e r é o número de simulações cujos valores é maior ou igual ao valor da estatística observada. A expressão e a argumentação usada estava correta, e parte de certa polêmica

que ocorreu sobre a validade do expressão obtida decorreu, aparentemente, do fato de que eles usaram indevidamente a terminologia *não-tendencioso*, que é característico de métodos de estimação de parâmetros.

A estimativa para o valor do $p-$Valor pelo método de Monte Carlo também é realizada por Aldstadt (2007), em $p - Valor_{MC} = R/(M + 1)$, em que R é o *Rank* das permutações, sendo definido por *quantos valores do conjunto das estatísticas das permutações de Monte Carlo e da estatística observada são maiores ou iguais à própria estatística observada*, e M é igual ao número de permutações de Monte Carlo. Também no código Knox de Meyer (2017), e na fórmula de Tango (2010), usa-se a mesma abordagem.

Nota-se, por fim, que o cálculo do $p-$Valor pelo método de Monte Carlo é sensível ao método de amostragem e os valores aleatórios gerados à amostragem em si. Assim sendo, distintas execuções geram diferentes variações no resultado. Uma solução à minimização da variabilidade do valor simulado é aumentar o valor de N_{rep}, visto que, estatisticamente, ele tende a se estabilizar quando o número de simulações é suficientemente grande. A amostragem dos casos de entrada é feita sob os tempos de ocorrência, mantendo as localizações espaciais fixas (TANGO, 2010).

O cálculo da distância espacial é realizado por meio do algoritmo à distância euclidiana, determinada pela expressão (2.18). Analogamente, o cálculo da distância temporal é realizado pela diferença absoluta entre dois valores, determinada pela expressão (2.19). Considerando que d_{ij} denota a métrica para as distâncias entre os casos i e j, é usual denotar a distância espacial por d_{ij}^S, e a distância temporal por d_{ij}^T.

$$d_{ij}^S = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \tag{2.18}$$

e

$$d_{ij}^T = |t_i - t_j| \tag{2.19}$$

Para efeito dos cálculos operacionais, para as coordenadas espaciais necessita-se que os dados estejam no formato *UTM - Universal Transversa de Mercator* - e para as temporais, no formato "Data Juliana". E para transformar as datas de entrada para o formato Juliano, inicialmente no formato "dia/mês/ano", utilizou-se a função "julday", retirado de Press et al. (1996).

2.1.2 Exemplo

Para ilustrar a abordagem realiza-se um exemplo completo do teste de Knox, utilizando dados adaptados de Boza (2009) como apresentado na Tabela (2.2).

Tabela 2.2: Exemplo recortado de Boza (2009).

Caso	Data	X	Y
1	05/01/1997	1	1
2	05/01/1997	1	4
3	06/01/1997	3	2
4	07/01/1997	3	2
5	08/01/1997	2	1

Primeiramente, faz-se necessário calcular todas as combinações dos casos de entrada que, como se pode verificar utilizando (2.1), totalizam $N = (5(5-1))/2 = 10$. De todas as duplas combinadas, calculam-se as distâncias espaciais e temporais, que são explicitadas na Tabela (2.3). Nota-se que foram explicitadas apenas 10 duplas, enquanto 20 são possíveis, no entanto, as distâncias de i até j e de j até i são iguais, tornando desnecessária a consideração delas, tanto que as formulações 2.4, 2.8-2.11 incluem o fator $1/2$ para desconsiderar essa duplicidade.

Tabela 2.3: Distâncias calculadas entre as combinações dos casos, com espacial em "unidades de distância", *u.d.*, e temporal em dias.

Dupla	Casos	Espacial	Temporal
1	1 & 2	3,0	0
2	1 & 3	2,2	1
3	1 & 4	2,2	2
4	1 & 5	1,0	3
5	2 & 3	2,8	1
6	2 & 4	2,8	2
7	2 & 5	3,2	3
8	3 & 4	0,0	1
9	3 & 5	1,4	2
10	4 & 5	1,4	1

Calculadas as distâncias, comparam-se as distâncias de cada dupla com os limiares espaço-temporais. Então se, por exemplo, a distância espacial entre os casos 1 e 2 for menor que o limiar espacial estimulado, essa dupla é considerada próxima no espaço. Por motivos estritamente didáticos, os limiares estipulados são 1 u.d. e 1 dia, resulta na tabela (2.4). Observa-se que em

diferentes implementações, operadores relacionais distintos foram utilizados. Neste exemplo bem como em Tango (2010), as distâncias exatamente iguais aos limiares são consideradas próximas, de modo que $d_{ij} \leqslant \delta$, enquanto em Meyer (2017) são considerados distantes, de modo que $d_{ij} < \delta$.

Tabela 2.4: Adjacências de todas as duplas por extenso.

Dupla	Casos	Espacial	Temporal
1	1 & 2	Distante	Próxima
2	1 & 3	Distante	Próxima
3	1 & 4	Distante	Distante
4	1 & 5	Próxima	Distante
5	2 & 3	Distante	Próxima
6	2 & 4	Distante	Distante
7	2 & 5	Distante	Distante
8	3 & 4	Próxima	Próxima
9	3 & 5	Distante	Distante
10	4 & 5	Distante	Próxima

Para classificar as duplas, é conveniente expressá-las em termos de a_{ij}^S e a_{ij}^T, como na Tabela (2.5), que apresenta uma coluna adicional denotada por "Ambas" representando duplas que estão próximas no espaço e no tempo simultaneamente, além dos totais parciais das últimas três colunas. A terceira coluna pode ser obtida por $a_{ij}^S \cdot a_{ij}^T$, e os três totais parciais podem ser substituídos por n_s, n_t e n_{st}, respectivamente.

Tabela 2.5: Adjacências de todas as duplas por extenso e com totais parciais, em termos de a_{ij}^S e a_{ij}^T.

Dupla	Casos	Espacial	Temporal	Ambas
1	1 & 2	0	1	0
2	1 & 3	0	1	0
3	1 & 4	0	0	0
4	1 & 5	1	0	0
5	2 & 3	0	1	0
6	2 & 4	0	0	0
7	2 & 5	0	0	0
8	3 & 4	1	1	1
9	3 & 5	0	0	0
10	4 & 5	0	1	0
Total		2	5	1

Com tais adjacências, pode-se realizar a categorização para a Tabela de contingência, como

na tabela (2.6).

Tabela 2.6: Tabela de contingência do exemplo.

Temporal \ Espacial	Próximos	Distantes	Total
Próximos	1	4	5
Distantes	1	4	5
Total	2	8	10

Com os resultados até então obtidos, pode-se calcular a esperança matemática da estatística de Knox, n_{st}, utilizando a expressão (2.5), que resulta em $E(n_{st}) = 2n_s n_t / n(n-1) = 2 \cdot 2 \cdot 5/5(4) = 20/20 = 1$ para o exemplo e, consequentemente, o cálculo do $p - Valor_{Poisson}$ utilizando (2.14), resulta em $p - Valor_{Poisson} = Pr(Poisson(E(n_{st})) \leqslant n_{st}) = Pr(Poisson(1) \leqslant 1) = 0,73576$. Para o cálculo da variância, por sua vez, deve-se obter primeiramente os valores para n_{2s} e n_{2t}, por (2.10) e (2.11) respectivamente, removendo a divisão por dois de ambas por não realizar a contagem de casos duplicados, como dito anteriormente. Assim, com $n_{2s} = 0$, $n_{2t} = 2$, obtém-se que

$$\begin{aligned}
\xi(n_{st}^2) &= \frac{2n_s n_t}{n(n-1)} + \frac{4n_{2s}n_{2t}}{n(n-1)(n-2)} + \frac{4\left[n_s(n_s-1)-2n_{2s}\right]\left[n_t(n_t-1)-2n_{2t}\right]}{n(n-1)(n-2)(n-3)} \\
&= \frac{2\cdot 2\cdot 5}{5(5-1)} + \frac{4\cdot 0\cdot 2}{5(5-1)(5-2)} + \frac{4\left[2(2-1)-2\cdot 0\right]\left[5(5-1)-2\cdot 2\right]}{5(5-1)(5-2)(5-3)} \\
&= \frac{20}{20} + \frac{0}{60} + \frac{4\left[2\right]\left[16\right]}{120} \\
&= 1 + 0 + \frac{128}{120} \\
&= 2,066 \\
Var(n_{st}) &= \xi(n_{st}^2) - \xi(n_{st})^2 \\
&= 2,0666 - 1^2 = 1,0666
\end{aligned} \tag{2.20}$$

Para comparação, utilizando a expressão (2.7) ao invés de (2.6) tem-se

$$
\begin{aligned}
\xi(n_{st}^2) &= \frac{2n_s n_t}{n(n-1)} + \frac{4n_{2s}n_{2t}}{n(n-1)(n-2)} + \frac{4\left[n_s(n_s-1)-n_{2s}\right]\left[n_t(n_t-1)-n_{2t}\right]}{n(n-1)(n-2)(n-3)} \\
&= \frac{2\cdot 2\cdot 5}{5(5-1)} + \frac{4\cdot 0\cdot 1}{5(5-1)(5-2)} + \frac{4\left[2(2-1)-0\right]\left[5(5-1)-2\right]}{5(5-1)(5-2)(5-3)} \\
&= \frac{20}{20} + \frac{0}{60} + \frac{4\left[2\right]\left[18\right]}{120} \\
&= 1 + 0 + \frac{144}{120} \\
&= 2,2 \\
Var(n_{st}) &= \xi(n_{st}^2) - \xi(n_{st})^2 \\
&= 2,2 - 1^2 = 1,2
\end{aligned}
\tag{2.21}
$$

Para as simulações de Monte Carlo, deve-se realizar a amostragem sem repetição dos casos. A Tabela (2.7) apresenta os cálculos de 9 amostragens e do valor inicial de n_{st}, considerando que as todas as nove simulações estão explícitas no Apêndice A.

Tabela 2.7: Simulações de Monte Carlo, onde 0 é a inicial.

Simulação	n_{st}
0	1
1	0
2	1
3	2
4	1
5	0
6	2
7	1
8	1
9	1

Para o cálculo do valor de $p - Valor_{MC}$, utiliza-se (2.16), com $N_{rep} = 9$ e $R = 8$, considerando que a simulação 0 já é contada na computação de R com o valor "$+1$"explícito na expressão (2.17), conforme discutido na Seção (2.1.1). Isto é, das simulações 1 a 9 contam-se quantas são maiores ou iguais ao valor inicial de n_{st} e incrementa-se por um o valor no final, como pode ser visto na tabela 2.8. Finalmente, tem-se que $p - Valor_{MC} = (7+1)/(9+1) = 8/10 = 0,8$.

Tabela 2.8: Contagem acumulativa *rank* das simulações de MC do exemplo

Simulação	Rank
1	1
2	1
3	2
4	3
5	4
6	4
7	5
8	6
9	7

Todos os resultados obtidos no exemplo podem ser sumarizados na Tabela (2.9).

Tabela 2.9: Resultados obtidos no exemplo.

Variável	Valor
n_s	2
n_t	5
n_{st}	1
$E(n_{st})$	1
$p - Valor_{Poisson}$	0,73576
n_{2s}	0
n_{2t}	1
$Var(n_{st})$	1,0666
$p - Valor_{MC}$	0,8

Com esses valores conclui-se que o número esperado de pares próximos no espaço e no tempo é igual a 1 e que a probabilidade de se observar pelo menos 1 caso é de 0,7657, ao utilizar o método de Poisson, ou de 0,8, ao utilizar o método de Monte Carlo.

2.2 Teste de Knox Incremental

Como em muitas aplicações de análise de clusterização, o intervalo temporal não deve ser acumulativo, mas considerar apenas o intervalo de tempo de ocorrência entre sucessivos eventos, Aldstadt (2007) desenvolveu uma formulação baseada no conceito de intervalo de ocorrência do evento, o serial interval, que corresponde a análise realizada pelo autor, ao tempo entre ocorrência de sucessivos casos de uma doença de cunho epidemiológico.

Nessa abordagem, a Significância da clusterização é verificada com simulações de Monte Carlo (ALDSTADT, 2007). E na pesquisa epidemiológica, o *serial interval* é utilizado para

descobrir a origem de um patógeno e a taxa de espalhamento no espaço e no tempo. Além de seu uso na etiologia, este parâmetro é vital à avaliação do risco de uma doença ou de tentativas de interrompimento de sua transmissão. Também é utilizado em modelos matemáticos de doenças utilizados para testar a eficácia de métodos de controle (ALDSTADT, 2007).

O método incremental de Aldstadt (2007) baseia-se no teste de Knox que toma como pressuposto o fato de que a transmissão de uma doença contagiosa terá clusterização espacial devido a maior probabilidade de contato com indivíduos próximos. Ao examinar casos dentro de um intervalo que estão mais próximos temporalmente do que se espera em uma situação em que não há um processo infeccioso, há grande probabilidade de a transmissão estar contida no intervalo estudado.

O tratamento do iKnox, chamado assim da junção de "incremental" e "Knox", é determinar o *serial interval* de uma doença infecciosa, sem a necessidade de dados adicionais a não ser aqueles utilizados nos estudos comuns da saúde pública (tupla data/local de ocorrência de um caso da doença sob estudo), testando diversos intervalos temporais ao invés de uma hipótese de interação espaço-temporal mais genérica. Ademais, a tendência de gerações individuais apresentarem uma clusterização próxima no tempo pode confundir tentativas de determinação do intervalo com testes acumulativos, como o caso de Baker (1996) (ALDSTADT, 2007).

Assim, para determinar a estatística do teste incremental, deve-se alterar a expressão (2.3) para (2.22).

$$b_{ij}^{T} = \begin{cases} 1, \text{casos } i \; e \; j \text{ estão separados por um intervalo de tempo } t \\ 0, \text{caso contrário} \end{cases} \tag{2.22}$$

Se o intervalo t for do tipo h, ao invés de algo como $h_1 \leqslant d_{ij}^{T} \leqslant h_2$, pode-se reescrever a expressão anterior como em (2.23)

$$b_{ij}^{T} = \begin{cases} 1, i \neq j \text{ e } d_{ij}^{T} = h \\ 0, \text{caso contrário} \end{cases} \tag{2.23}$$

em que os casos i e j estão separados por um tempo exatamente igual à h, como por exemplo, h dias, de modo que a Estatística de Knox incremental é calculada como em (2.24).

$$IK_{st} = \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} a_{ij}^{S} b_{ij}^{T} \tag{2.24}$$

Observe-se que método Knox usual emprega tanto $i \neq j$ e $d_{ij}^T \leqslant h$ quanto $i \neq j$ e $d_{ij}^T < h$, dependendo da implementação. O iKnox, no entanto, tem que os casos i e j são próximos se estão separados por um intervalo de tempo. Se, por exemplo, tal intervalo for de 5 dias, resulta que $1, se\ i \neq j$ e $d_{ij}^T = 5$.

Como a metodologia é utilizada com um intervalo de parâmetros críticos espaciais e temporais, vários testes são inerentes ao enfoque iKnox. Uma maneira de representar os resultados obtidos é a noção de Risco Elevado (*Elevated Risk, ER*), obtida, para cada caso de variação dos parâmetros críticos espaciais e temporais como na expressão (2.25).

$$ER = \frac{IK_{st}}{\mu_{MC}} \tag{2.25}$$

em que IK_{st} é a estatística do teste incremental e μ_{MC} é a média das estatísticas das permutações de Monte Carlo.

É interessante observar que existem diferentes formas à apresentação da Distribuição Z para o caso de dados observados, Z_{IK}. A expressão empregada neste trabalho é como em (2.26), que é a mesma de Aldstadt (2007), com a notação modificada.

$$Z_{IK} = \frac{IK_{st} - \mu_{MC}}{SE_{MC}} \tag{2.26}$$

em que $SE_{MC} = SD_{MC}/\sqrt{n}$, sendo SE_{MC} o erro padrão e SD_{MC} o desvio padrão das estatísticas das permutações de MC, respectivamente.

Também é relevante destacar que na determinação de proximidade espaço temporal, por meio da Estatística Observada no método de Aldstadt (2007), considera-se a descrição dada pelo autor, traduzida, "a_{ij}^s *é igual à* 1 *se os casos* i *e* j *estão próximos no espaço e* 0 *caso contrário,* a_{ij}^t *é igual à* 1 *se os casos* i *e* j *são próximos no tempo e* 0 *caso contrário, se* s *e* t *representam distâncias espaciais e temporais pré-definidas*".

2.2.1 Exemplo

De modo geral, o teste iKnox pode ser observado como várias execuções do teste de Knox com diferentes valores para os limiares, criando uma matriz de resultados, com uma dimensão

para variação do limiar espacial e outra para o temporal, por conseguinte, este exemplo baseia-se nos mesmos dados utilizados no exemplo da Seção 2.1.2.

Com o intuito de exemplificação puramente didática, estipulam-se as faixas de limiares críticos como de $\delta_S = \{0,5; 1; 1,5\}$ e $\delta_T = \{1; 2; 3\}$. Estipulados os conjuntos, realizam-se testes com todas as combinações de limiares entre os conjuntos, ou seja, o conjunto de parâmetros críticos pode ser escrito como $\delta_S \times \delta_T = \{(0,5; 1); (0,5; 2); (0,5; 3); (1,5; 1); (1; 2); (1; 3); (1,5; 1); (1,5; 2); (1,5; 3)\}$, é representado na Tabela (2.10).

Tabela 2.10: Limiares críticos para testes do iKnox.

Teste	δ_S	δ_T
1	0,5	1
2	0,5	2
3	0,5	3
4	1	1
5	1	2
6	1	3
7	1,5	1
8	1,5	2
9	1,5	3

Com o conjunto de limiares de teste, realiza-se um procedimento similar ao da Seção 2.1.2 dependendo de quais resultados são desejados e qual formulação é escolhida para eles. A fim de representar os dados apenas na noção de Risco Elevado, apresentado em (2.25), precisa-se somente dos valores de n_s, n_t e n_{st} para todos os testes, visto que o cálculo de ER requer tais termos. Na representação da Distribuição Z, por sua vez, é necessário o cálculo do desvio padrão. Observa-se que, como foi empregado o uso da expressão (2.26), precisa-se calcular a média dos valores simulados com Monte Carlo e o desvio padrão em relação aos mesmos valores.

A Tabela (2.11) apresenta os resultados para dos testes e a Tabela (2.12) apresenta os valores de Risco Elevado codificados com tons de cinza. Nota-se que todas as nove simulações de MC de cada um dos nove testes estão explícitas no Apêndice A.

Tabela 2.11: Resultados dos testes de iKnox de exemplo.

Teste	n_{st}	$E(n_{st})$	ER	μ_{MC}	SD_{MC}	Z_{IK}
1	1	0,4	2,5	0,4	0,516	2,598
2	0	0,3	0	0,3	0,483	-1,388
3	0	0,2	0	0,4	0,422	-2,121
4	1	0,8	1,25	0,8	0,632	0,707
5	0	0,6	0	0,6	0,692	-1,919
6	1	0,4	2,5	0,4	0,516	2,598
7	2	1,6	1,25	1,6	0,516	1,732
8	1	1,2	0,83	1,2	0,421	-1,06
9	1	0,8	1,25	0,888	0,333	0,745

Tabela 2.12: Risco Elevado dos testes, onde o risco é crescente das cores mais claras para mais escuras, e os números são referentes aos valores de ER.

3	0	2,5	1,25
2	0	0	0,83
1	2,5	1,3	1,3
Dias / *u.d.*	0,5	1,0	1,5

Com esses valores conclui-se que o maior risco de clusterização espaço-temporal dos casos de teste está quando estes estão separados por 1 dia e 0,5 *u.d.* e por 3 dias e 1 *u.d.*.

Capítulo 3

Implementações e otimizações sobre o Teste de Knox

Apesar do teste de Knox ser bastante utilizado para a clusterização espaço-temporal, não há uma ampla variedade de implementações. Duas que podem ser destacadas são a de Tango (2010) e a de Höhle et al. (2017), onde a primeira apresenta a possibilidade de cálculo dos termos n_2, isto é, n_{2s} e n_{2t}, mas é menos eficiente, tanto na complexidade temporal quanto na espacial, em relação à segunda.

A implementação de Tango (2010) é dividida em duas funções, onde cada uma traz resultados semelhantes entre si, mas com base em métodos diferentes. A primeira é "KnoxA.test", que traz a estatística do método, média, variância, estatística normalizada e valores de $p - Valor$ simulados. A segunda, chamada "KnoxM.test" traz a estatística, um vetor com todos as estatísticas simuladas por Monte Carlo e o $p - Valor$ simulado. Ambas tem como entrada três vetores, coordenadas X, Y, tempo, e dois parâmetros críticos. A função que utiliza MC tem um parâmetro adicional para a quantidade de replicações.

Escolhas de implementação que levam a um aumento na complexidade espacial (memória) de ambas as funções podem ser atribuídas a o uso de matrizes $n \times n$, com n sendo o número de casos de entrada, com valores de 64 bits para armazenar as distâncias das duplas e de matrizes com valores de 32 bits para armazenar a adjacência das duplas testadas. Então para um teste com 1.000 casos de entrada, por exemplo, são necessárias duas matrizes 1.000×1.000 de 64 bits, totalizando $16.000.000$ bytes, para as distâncias e duas 1.000×1.000 de 32 bits, totalizando $8.000.000$ bytes, para as adjacências, considerando que estas são mantidas em memória até término da execução.

Já as escolhas que impactam na complexidade temporal (custo computacional), incluem o cálculo de distâncias repetidas para ambas as funções, isto é, as distâncias das duplas (i, j) e (j, i) são iguais e são calculadas explicitamente; o método para cálculo dos termos n_{2s} e n_{2t} que possui três laços aninhados, essencialmente tornando-o $O(n^3)$, na função "KnoxA.test" , e a repetição do cálculo de todas as distâncias temporais, incluindo o cálculo duplicado supracitado, para cada simulação de MC, na função "KnoxM.test". Nota-se que, embora essa implementação seja razoável para testes com n pequeno, até por volta de $n = 100$, ela não é adequada para n grande, e não apresenta nenhum tipo paralelismo.

A implementação de Höhle et al. (2017), por sua vez, apresenta ambas as funções compactadas em uma, com o usuário escolhendo entre elas por meio de parâmetros. Se a escolha for de obter $p - Valor$ aproximado, a etapa de MC não é executada e são retornados, além deste último, a estatística do método e a média, não trazendo a variância e nem a estatística normalizada. Se for escolhido o $p - Valor$ simulado, executa-se a etapa de MC, retornando os mesmos valores que a função equivalente do autor anterior. Outra diferença está nas entradas, que são vetores de distâncias já calculadas, ao invés de vetores com os casos em si, apesar disso, o custo do cálculo das distâncias deve ser considerado pois é inerente à obtenção de resultados.

Em relação à implementação de Tango (2010), a complexidade de espaço é reduzida para menos da metade devido ao cálculo não repetido de duplas, em outras palavras, para n casos de entrada, este armazena $N = (n(n-1))/2$ ao invés de n^2. Enquanto a função não calcula as distâncias internamente, ela recebe ambos os vetores por parâmetro, indicando que não há melhoria no aspecto de manter os vetores de adjacência e distâncias ao mesmo tempo em memória. Para exemplificar, em um teste de 1.000 entradas, essa função teria dois vetores de distâncias com tamanho 999.000 de 64 bits, totalizando $7.992.000$ bytes, e dois vetores de adjacências de 32 bits, totalizando $3.996.000$ bytes.

A complexidade de tempo também apresenta melhorias, mas às custas de não obtenção de variância ao se utilizar $p - Valor$ aproximado, visto que este trecho é completamente ausente nessa implementação, possivelmente porque a tradução direta das expressões (2.10) e (2.11) para código sugere o uso de matrizes, o que impacta na complexidade do espaço, e do tempo, se as posições duplicadas tiverem as distâncias calculadas novamente. No trecho de MC, também há melhorias de complexidade, uma vez que as distâncias não são calculadas para cada nova

simulação, apenas a amostragem dos vetores de adjacência. Diferentemente do outro caso e embora que somente no trecho das simulações de MC, há paralelismo. Enquanto este último melhora o tempo de execução, traz impacto negativo no uso de memória, já que para cada *thread*, há uma cópia dos vetores de adjacência, que devem ser amostrados.

A implementação proposta neste trabalho apresenta um misto das duas citadas anteriormente, onde há possibilidade do cálculo da variância associado ao $p - Valor$ aproximado, mesmo com as melhorias de espaço e tempo apresentadas na segunda abordagem. Ainda há o uso de paralelismo com o enfoque de melhoria no tempo de execução, semelhante ao segundo exemplo, sem a desvantagem no aumento do uso de memória.

No quesito de espaço, a implementação apresentada posteriormente é similar ao segundo exemplo quando se considera a quantia de duplas armazenadas, no entanto, cada adjacência ocupa menos espaço na ordem de 32 vezes menor, já que 4 bytes de um inteiro em R contra 1 bit. Enquanto as distâncias possuem o mesmo tamanho de ambos os exemplos de implementações e a quantia de duplas do segundo exemplo, há melhorias em relação à quantas estão presentes em memória ao mesmo tempo. Seguindo a linha com um teste de 1.000 entradas, utilizariam-se $7.992.000$ bytes para as distâncias e 249.750 bytes para as adjacências, se todas estivessem em memória ao mesmo tempo.

Sobre a complexidade de tempo, ela é similar com o segundo caso quando escolhido o $p - Valor$ simulado, mas com este aproximado, há melhorias na etapa do cálculo dos termos n_{2s} e n_{2t} reduzindo para dois laços aninhados, essencialmente tornando-o $O(n^2)$. Como mencionado anteriormente, o paralelismo no trecho de MC acontece internamente nas estruturas de adjacência, não sendo necessário ter uma para cada *thread*, mantendo um desempenho semelhante no tempo, mas sem grandes perdas de espaço inerentes à outra abordagem.

3.1 Algoritmo e Fluxograma Geral

O Fluxograma da Figura 3.1 e o Algoritmo 1 referem-se ao fluxo de execução geral do algoritmo que implementa do método de Knox. Neste algoritmo, é calculada a Estatística Observada, os parâmetros n_s e n_t e as matrizes de adjacência temporal e espacial. Com as matrizes de adjacência pode-se calcular os termos n_{2s} e n_{2t} e realizar as simulações de Monte Carlo. Cada uma dessas etapas, individualmente, é detalhada em sequência nesta Seção. Observa-se

que o modelo dos fluxogramas é explicitado no Apêndice B.

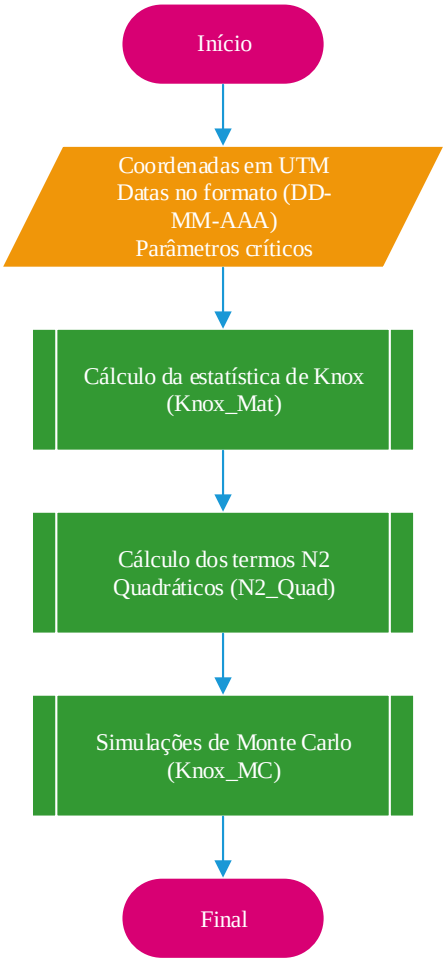

Figura 3.1: Fluxograma geral do algoritmo de Knox.

Note que no Algoritmo 1 Mat_S e Mat_T denotam, respectivamente, as matrizes de adjacência espacial e temporal.

Algoritmo 1: ALGORITMO GERAL DE KNOX NÃO PARALELO

início
  \\Dados de entrada em UTM, para espaço, e DD-MM-AAAA, para tempo
  Knox-Mat($Crit_S$, $Crit_T$, Casos-Entrada) \\Saída: Mat_S, Mat_T, $Knox_{Obs}$
  N2-Quadrático(Mat_S, Mat_T) \\Saída: n_{2s} e n_{2t}
  Monte-Carlo(Mat_S, Mat_T) \\Saída: $pValor_{MC}$, $Mean_{MC}$
fim

O cálculo da Estatística de Knox precisa das adjacências para todas combinações dos casos. Para isso, calculam-se as distâncias e comparam-se elas com os limiares críticos, como mostrado no Fluxograma da Figura 3.2 e no Algoritmo 2.

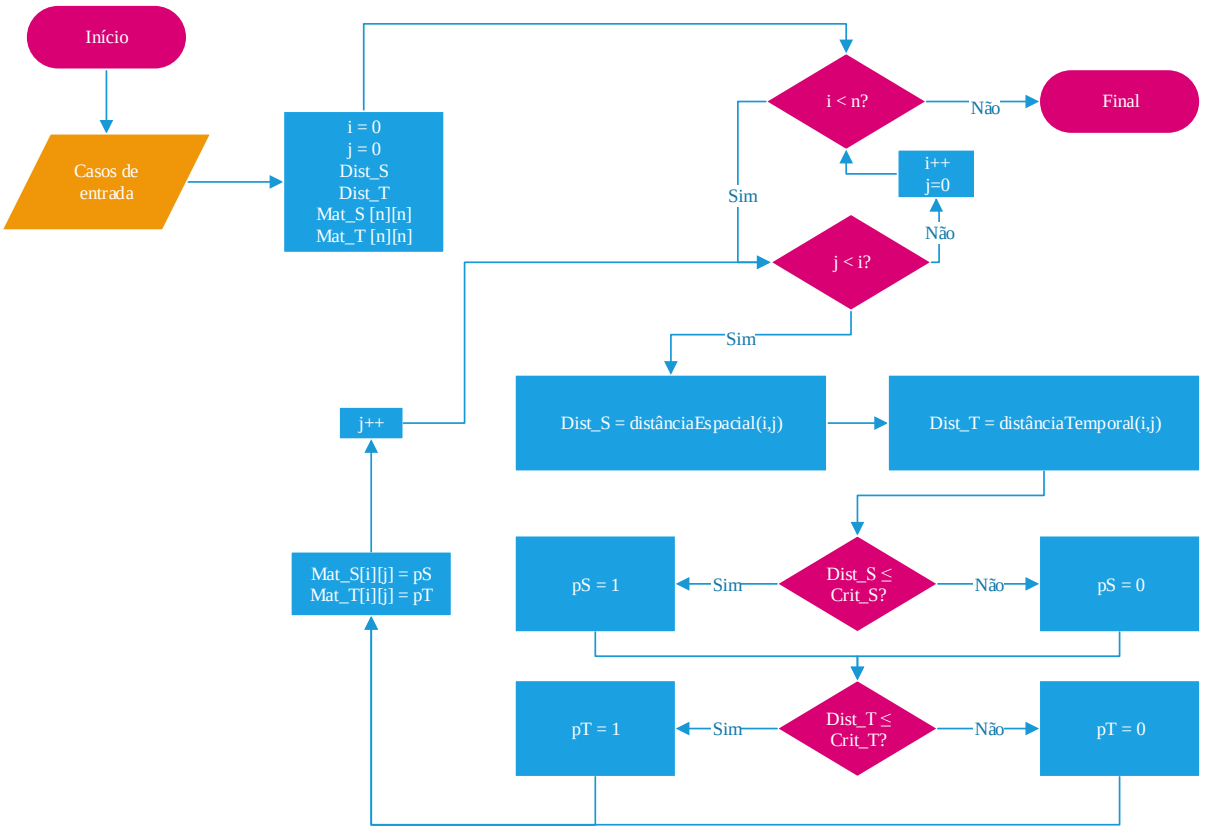

Figura 3.2: Fluxograma do cálculo das adjacências de todas as combinações entre os casos de entrada.

Algoritmo 2: CÁLCULO DAS ADJACÊNCIAS DE TODAS AS COMBINAÇÕES ENTRE OS CASOS DE ENTRADA

```
1  início
2      \\Array de n elementos indexado de 0 até n-1
3      n_st = 0
4      n_s = 0
5      n_t = 0
6      para i de 0 até n-1 faça
7          para j de 0 até i-1 faça
8              Dist_S = distânciaEspacial(i,j)
9              Dist_T = distânciaTemporal(i,j)
10
11             se Dist_S ⩽ Crit_S então
12                 pS = 1
13             senão
14                 pS = 0
15             fim
16             se Dist_T ⩽ Crit_T então
17                 pT = 1
18             senão
19                 pT = 0
20             fim
21
22             n_st+ = pS · pT
23             n_s+ = pS
24             n_t+ = pT
25
26             Mat_S[i][j] = pS
27             Mat_T[i][j] = pT
28         fim
29     fim
30 fim
```

Dadas as adjacências, pode-se calcular a estatística do método, os termos n_s e n_t e os termos n_{2s} e n_{2t}. O Fluxograma da Figura 3.3 e o Algoritmo 3 mostram como é realizado o cálculo para os três primeiros, enquanto o cálculo dos dois últimos é mostrado posteriormente. Para uma simulação de Monte Carlo, realiza-se uma amostragem sobre a matriz de adjacência temporal, Mat_T nos algoritmos e Mat_T nos fluxogramas, e então utiliza-se o mesmo algoritmo para a estatística do método. Para cada nova simulação, uma nova amostragem da matriz temporal é realizada.

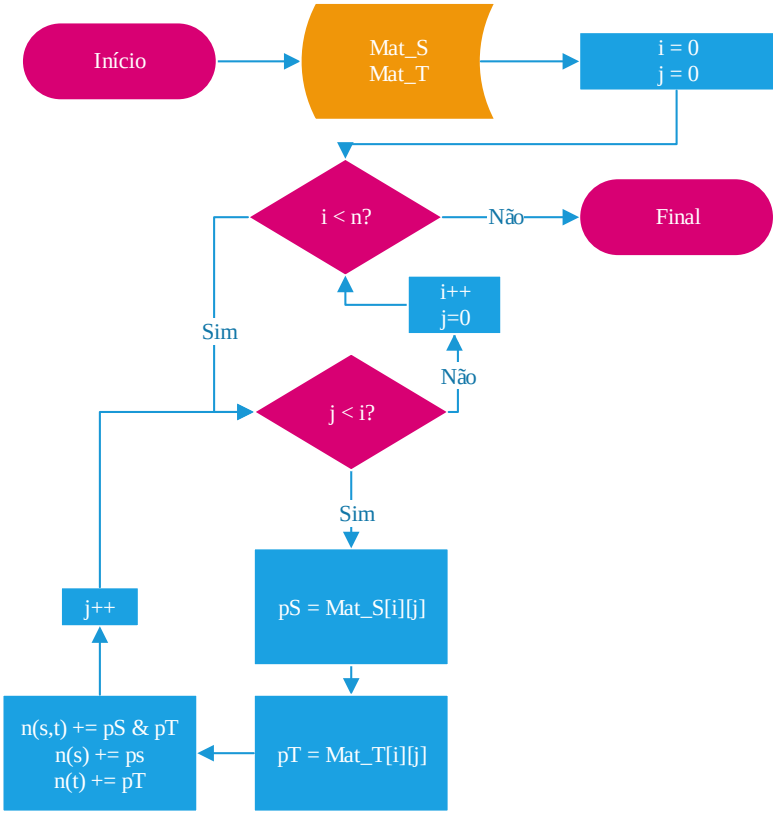

Figura 3.3: Fluxograma do cálculo de n_{st} com matrizes de adjacência de entrada.

Algoritmo 3: ESTATÍSTICA DE KNOX COM MATRIZ DE ADJACÊNCIA DE ENTRADA

início
  \\Array de n elementos indexado de 0 até n-1
  $n_{st} = 0$
  $n_s = 0$
  $n_t = 0$
  para *i de 0 até n-1* **faça**
    para *j de 0 até i-1* **faça**
      $pS = Mat_S[i][j]$
      $pT = Mat_T[i][j]$

      $n_{st} += pS \cdot pT$
      $n_s += pS$
      $n_t += pT$
    fim
  fim
fim

3.2 Cálculo da Média e da Variância da Aproximação de Poisson

Como supracitado, com as matrizes de adjacências pode-se obter os termos n_{2s} e n_{2t}. A abordagem para tal utilizada por Tango (2010) tem uma complexidade assintótica de $O(n^3)$, que foi reduzida no algoritmo implementado neste trabalho para $O(n^2)$. Os respectivos Fluxograma da Figura 3.4 e Algoritmo 4 são como:

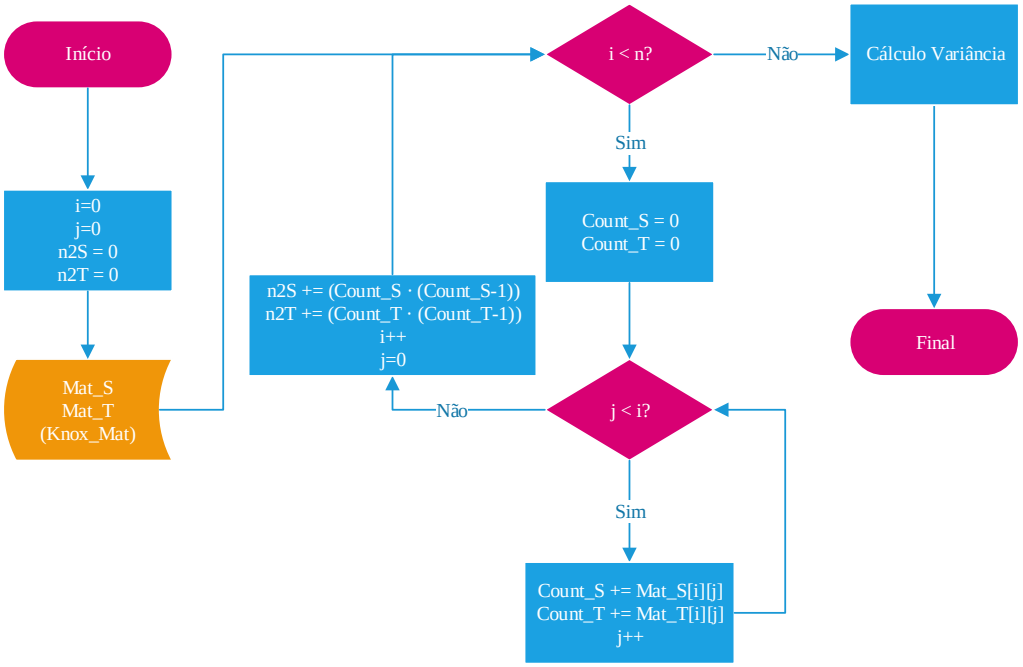

Figura 3.4: Cálculo dos termos n_{2s} e n_{2t} com complexidade quadrática.

Algoritmo 4: CÁLCULO DOS TERMOS n_2 COM COMPLEXIDADE QUADRÁTICA.

```
início
    \\Array de n elementos indexado de 0 até n-1
    n_2s = 0
    n_2t = 0
    para i de 0 até n-1 faça
        Count_S = 0
        Count_T = 0
        para j de 0 até i-1 faça
            Count_S+ = Mat_S[i][j]
            Count_T+ = Mat_T[i][j]
        fim
        n_2s+ = (Count_S · (Count_S − 1))
        n_2t+ = (Count_T · (Count_T − 1))
    fim
fim
```

Como a otimização do Algoritmo 4 em relação ao de Tango (2010) foi obtida conforme mostrada no exemplo abaixo, considerando-se que dado uma matriz de adjacência como na Figura 3.5, podemos calcular o termo n_2 para tal matriz, que se for matriz de adjacência espacial, calcula-se o termo n_{2s}, e se for a temporal, calcula-se n_{2t}.

	j	0	1	2	3	4	5
i							
0		0	1	1	1	1	0
1		1	0	1	0	1	0
2		1	1	0	0	0	1
3		1	0	0	0	0	1
4		1	1	0	0	0	0
5		0	0	1	1	0	0

Figura 3.5: Matriz de adjacência considerada para desenvolver o exemplo apresentado em sequência.

Com o código de complexidade assintótica cúbica deve-se realizar, consequentemente, n^2 operações por linha. Com cada elemento, que uma posição i, j da matriz, tem-se que realizar $n - 1$ operações, em que cada uma delas é uma multiplicação entre dois elementos que são ora iguais à 0 ora iguais à 1.

O laço mais interno do algoritmo faz, em essência, a multiplicação da posição i, j atual

da matriz de adjacência com todos os outros elementos da mesma linha, excluindo o próprio elemento. Assim, pode-se explicitar todas as iterações sobre uma mesma linha, como na Figura 3.6.

i	j	k	mat_ij	mat_ik	n2	i	j	k	mat_ij	mat_ik	n2	i	j	k	mat_ij	mat_ik	n2
0	0	0	0	0	0	0	2	0	1	0	3	0	4	0	1	0	9
0	0	1	0	1	0	0	2	1	1	1	4	0	4	1	1	1	10
0	0	2	0	1	0	0	2	2	1	1	4	0	4	2	1	1	11
0	0	3	0	1	0	0	2	3	1	1	5	0	4	3	1	1	12
0	0	4	0	1	0	0	2	4	1	1	6	0	4	4	1	1	12
0	0	5	0	0	0	0	2	5	1	0	6	0	4	5	1	0	12
0	1	0	1	0	0	0	3	0	1	0	6	0	5	0	0	0	12
0	1	1	1	1	0	0	3	1	1	1	7	0	5	1	0	1	12
0	1	2	1	1	1	0	3	2	1	1	8	0	5	2	0	1	12
0	1	3	1	1	2	0	3	3	1	1	8	0	5	3	0	1	12
0	1	4	1	1	3	0	3	4	1	1	9	0	5	4	0	1	12
0	1	5	1	0	3	0	3	5	1	0	9	0	5	5	0	0	12

Figura 3.6: Iterações sob as posições i, j da matriz de exemplo.

As regiões marcadas com verdes são posições onde há um incremento para o termo n_2 em questão, ou n_{2s} ou n_{2t} e as regiões alaranjadas são desconsideradas visto que representam a multiplicação pelo próprio elemento da posição i, j. Assim, são realizadas efetivamente $n - 1$ operações sobre cada tupla i, j, visto que uma sempre é ignorada.

Como cada iteração realiza somente uma multiplicação entre duas variáveis que podem assumir dois valores, ou 0 ou 1, toda iteração em que o valor i, j for igual à 0, não haverá nenhum impacto no valor de n_2, podendo, portanto, ser ignorada. Em outras palavras, cada iteração realiza uma operação lógica and entre dois valores, quando o elemento mais à esquerda do operador for falso ou 0 a expressão não precisa ser avaliada, visto que seu resultado só pode ser igual a falso, que é uma avaliação em curto-circuito, não sendo acrescentado ao valor de n_2.

Ao observar o código em si, mostrado no Algoritmo 5, nota-se que, dentro do laço mais interno, o valor de $a_{i,j}$ não muda, tornando possível uma otimização em que toda vez $a_{i,j}$ for igual à 0 tal laço não é executado. Considerando a linha 5 da matriz de exemplo em um algoritmo com essa otimização, são realizadas $2(n-1)$ operações, ao invés de $6(n-1)$.

Algoritmo 5: CÓDIGO DE COMPLEXIDADE CÚBICA PARA O CÁLCULO DE n_{2s} E n_{2t}

```
1  início
2      int i, j, k
3      int aij_s = 0, aij_t = 0
4      para i de 0 até Ncasos-1 faça
5          para j de 0 até Ncasos-1 faça
6              \\"_s"é em relação ao espacial e "_t"ao temporal
7              aij_S = Mat_S[i][j]
8              aij_T = Mat_T[i][j]
9              para k de 0 até Ncasos-1 faça
10                 se j ≠ k então
11                     n_2s+ = aij_S ∧ Mat_S[i][k]
12                     n_2t+ = aij_T ∧ Mat_T[i][k]
13                 fim
14             fim
15         fim
16     fim
17 fim
```

Uma característica que pode ser extraída da otimização anterior é que o número de vezes que o laço mais interno é executado é igual ao número de elementos 1 presentes naquela linha i. Assim, na linha 0 da matriz do exemplo, tal laço seria executado exatamente 4 vezes. Dentro desse contexto, o número de incrementos à variável n_2, n_{2s} ou n_{2t}, no laço mais interno, será igual ao número de elementos cujas posições na matriz são diferentes da atual, que é a condicional em que $j \neq k$, e que seus valores sejam iguais a 1.

Na Figura 3.6, essas situações estão destacadas em verde. Observa-se que o número de posições destacadas é exatamente igual ao número de $1s$ na linha menos 1, isto é, o número de total de incrementos realizados no laço mais interno é igual à $x - 1$, em que x é o número de $1s$ na linha. Como se sabe da otimização anterior que o número de vezes que este laço é executado é igual a x, pode-se determinar o incremento à n_2 de cada linha realizando-se uma contagem em cada linha e utilizando a expressão $x(x - 1)$ para determinar este incremento.

Em outras palavras, para cada linha i da matriz original, tem-se uma matriz $n \times n$ auxiliar, em que n é o número de casos, de $0s$ ou de $1s$; cada linha j da matriz auxiliar é completada com $0s$ se a posição i, j da matriz original for 0 ou com $1s$ caso contrário; as posições na diagonal principal da matriz auxiliar são zeradas, em que $j = k$, independentemente do valor

previamente colocado. A Figura 3.7 mostra como ficaria essa matriz auxiliar para a linha $i = 0$ da matriz original de exemplo.

		Linha i=0						
		0	1	1	1	1	0	4
	k	0	1	2	3	4	5	
j								
0		0	0	0	0	0	0	0
1		1	0	1	1	1	1	3
2		1	1	0	1	1	1	3
3		1	1	1	0	1	1	3
4		1	1	1	1	0	1	3
5		0	0	0	0	0	0	0
								12

Figura 3.7: Exemplo para a linha $i = 0$.

Com a matriz auxiliar montada, opera-se cada linha da mesma com a linha i de onde a matriz foi montada, $i = 0$, no exemplo, multiplicando posição a posição e acumulando em uma variável, uma para cada linha. Ao observar estas últimas, nota-se que o número de variáveis cujos valores diferem de zero é igual ao número de uns na linha atual, x, e que o valor nesses contadores é exatamente $x - 1$, resultando na mesma fórmula supracitada.

Com essa expressão, necessita-se, apenas, realizar uma contagem de $1s$ em cada linha de cada matriz de adjacência. Como para realizar essa contagem precisa-se comparar todos os valores em uma linha com n elementos e possui-se n linhas, a abordagem torna-se quadrática. Por fim, observa-se que se armazena metade das matrizes de adjacência, como uma otimização de espaço. Mesmo armazenando-se metade, a complexidade continua quadrática, pois para x permanecer igual, deve-se realizar a contagem em relação às linhas e em relação às colunas, ao invés de somente em relação às linhas, como apresentado na Figura 3.8

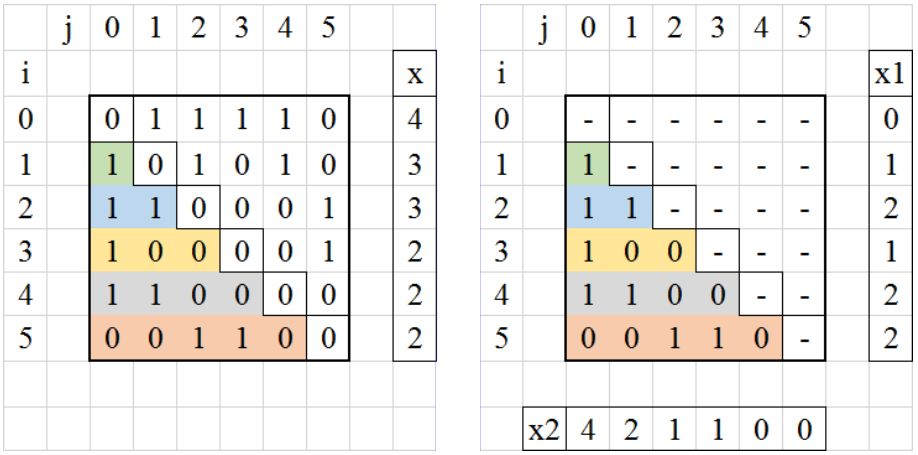

Figura 3.8: Esquerda: Vetor x com uma matriz de adjacência completa. Direita: Vetores montados com uma matriz de adjacência parcial. O vetor x é obtido como a soma dos vetores $x1$ e $x2$.

Considerando essas otimizações, têm-se duas abordagens em relação à complexidade de espaço para os cálculos dos termos n_2. Uma que requer que linhas inteiras estejam em memória antes de iniciar uma etapa do algoritmo, mas não requer nenhum contador a não ser os de resultado e outra que requer que exista um contador para cada linha, mas não necessita que as linhas estejam completamente em memória em nenhum intervalo de tempo. Ambas abordagens podem ser adaptadas tanto para operar com ora metade ora totalidade das matrizes de adjacência, cada uma com suas respectivas vantagens e desvantagens.

A primeira abordagem é a mostrada nas Figuras 3.6 e 3.7, en que realiza-se a contagem de $1s$ em uma linha, x, e incrementa-se o contador de resultado com $n_2 += x \cdot (x-1)$, de modo que ela requer no mínimo n elementos em memória ao mesmo tempo, como, por exemplo, ao manter a linha dois da matriz utilizada no exemplo anterior para calcular o x da mesma, visto na Figura 3.9.

	j	0	1	2	3	4	5
i							
0		0	1	1	1	1	0
1		1	0	1	0	1	0
2		1	1	0	0	0	1
3		1	0	0	0	0	1
4		1	1	0	0	0	0
5		0	0	1	1	0	0

Figura 3.9: Elemento mínimo para manter em memória no caso da abordagem sem contadores e com matriz de adjacência completa.

Pode-se estender este conceito para manter mais linhas em memória ao mesmo tempo, evitando intercalar várias etapas de processamento. Sendo b a quantia de linhas da matriz mantida em memória, temos que $1 < b \leqslant n$. Se essa abordagem for adaptada para matriz de adjacência triangular, deve-se manter os elementos refletidos em relação à diagonal principal da parte triangular que não está sendo usada, de modo que a mesma "linha" esteja presente em memória. Para exemplificar, observa-se a Figura 3.10, em que foi escolhido manter a triangular inferior em memória.

	j	0	1	2	3	4	5
i							
0		0	1	1	1	1	0
1		1	0	1	0	1	0
2		1	1	0	0	0	1
3		1	0	0	0	0	1
4		1	1	0	0	0	0
5		0	0	1	1	0	0

Figura 3.10: Elemento mínimo para manter em memória no caso da abordagem sem contadores e com matriz de adjacência parcial.

A segunda abordagem é similar ao visto nas partes esquerda e direita da Figura 3.8, em que tem-se um contador para cada linha/coluna, com armazenamento completo e parcial, respectivamente. Para o armazenamento da matriz completa, os contadores são atualizados conforme cada linha, como descrito anteriormente, no entanto, com a triangular, tem-se que realizar a

contagem em ambas as direções. A parte direita mostra um contador para cada linha e um para cada coluna, mas, ao comparar com a parte esquerda, nota-se que esses devem ser somados para obter o valor de x para o incremento ao resultado, portanto, a complexidade de espaço permanece a mesma do armazenamento completo, isto é, $O(n)$.

O crescimento do espaço necessário para as abordagens é estimado pelas equações (3.1) e (3.2), respectivamente. Na equação (3.2) há a diferenciação do fator de multiplicação conforme n pois um inteiro não sinalizado de 32 bits armazena, no máximo o valor 2^{32}, portanto, se $n > 65536$, contadores de mais de 32 bits são necessários, isto é, faz-se necessário o uso de inteiros não sinalizados de 64 bits. Observa-se que se os valores x forem próximos do mesmo limite, o uso de inteiros maiores também faz-se necessário, pois 32 bits não são suficientes para armazenar a soma dos contadores.

A partir das equações, nota-se que a complexidade de espaço da primeira abordagem é menor desde que $b < 32$, no entanto, esse número reduzido de linhas pode não aproveitar o desempenho disponível. Nos testes, um número de linhas que apresentou um desempenho satisfatório em diversos testes foi $n = 250$, fazendo com que em linhas tenha-se 62.500 bits e em contadores tenha-se 16.000 bits.

$$f(n) = x \cdot b \tag{3.1}$$

$$g(n) = \begin{cases} x \cdot 32, & se\ n \leqslant 2^{32} \\ x \cdot 64, & se\ n > 2^{32} \end{cases} \tag{3.2}$$

Assim sendo, com os termos n_s, n_t, n_{2s} e n_{2t} disponíveis, pode-se calcular a média e a variância conforme as Expressões (2.5) e (2.6), respectivamente, de David e Barton (1966).

3.3 Amostragem ao Cálculo das Simulações de Monte Carlo

Para a amostragem sem reposição dos elementos da matriz de proximidade temporal, utilizou-se o algoritmo de Fisher-Yates *shuffling*, como disponível em Knuth (1997), cujo Fluxograma é o da Figura 3.11 e o Algoritmo 6.

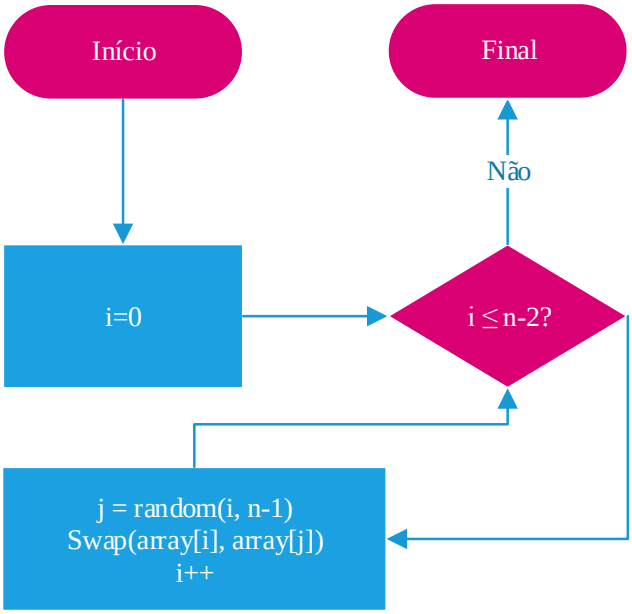

Figura 3.11: Fluxo do método de Fisher-Yates para amostragem sem reposição.

Algoritmo 6: AMOSTRAGEM FISHER YATES

início
  \\Array de n elementos indexado de 0 até $n-1$
  \\Para no intervalo $[0; n-2]$
  para *i de 0 até n-2* **faça**
    $j = random(i, n-1)$ \\Numero aleatorio no intervalo $[i$;n-1$]$
    $swap(array[i], array[j])$ \\Troca os elementos das posições i e j
  fim
fim

Observa-se que como é utilizada apenas metade da matriz para o cálculo de n_{st}, os valores são permutados apenas dentro da porção triangular utilizada. Isto é, se utilizada a triangular inferior, nenhum valor é permutado com outro pertencente à diagonal principal ou a triangular superior. Consequentemente, a primeira linha nunca passa pelo algoritmo por não possuir nenhum valor e a segunda por possuir apenas um valor.

Além de não permutar fora do recorte triangular, optou-se por permutar somente dentro da mesma linha na matriz, uma vez que a matriz não é gerada por inteiro em nenhum momento no tempo, devido a abordagem em blocos desenvolvida à otimização, como discutido na Seção 3.2. A Figura 3.12 ilustra essa situação.

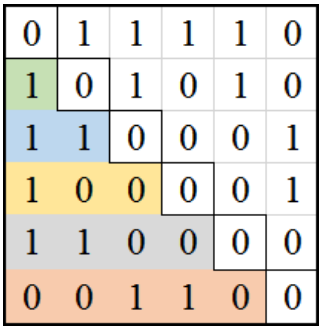

Figura 3.12: Esquema de particionamento para amostragem. Cada linha colorida, exceto a em verde, é passada ao algoritmo separadamente.

3.4 Estratégia em Blocos

Como visto na Seção 3.2 pode-se dividir o processamento das matrizes de adjacência em partes, para que as mesmas não precisem ser carregadas por completo em memória, sendo possível executar testes maiores. As estratégias discutidas a seguir são algumas das várias possibilidades existentes para realizar tal decomposição no processamento, sendo elas baseadas em matrizes. Outra estratégia, por exemplo, é empregando vetores como a utilizada em Meyer (2017), mas que pode ser mais trabalhosa para obter os termos n_2 necessários ao cálculo de Var por esse método. Para melhor detalhar nessa seção alguns aspectos da estratégia adotada, considera-se como exemplo a matriz apresentada na Figura 3.13.

	j	0	1	2	3	4	5	6	7	8	9
i											
0		0	1	1	0	0	0	1	0	0	1
1		1	0	0	1	1	0	0	0	1	1
2		1	0	0	0	1	0	1	1	0	1
3		0	1	0	0	1	1	0	1	1	0
4		0	1	1	1	0	0	1	0	0	0
5		0	0	0	1	0	0	1	0	1	0
6		1	0	1	0	1	1	0	0	1	1
7		0	0	1	1	0	0	0	0	0	0
8		0	1	0	1	0	1	1	0	0	1
9		1	1	1	0	0	0	1	0	1	0

Figura 3.13: Matriz de exemplo para a presente seção.

Conforme discutido anteriormente, há possibilidade de armazenar as matrizes de adjacência

completas ou parciais, associando-se ao algoritmo escolhido para o cálculo dos termos n_2. Se escolhido um algoritmo que não mantém contadores como discutido na Seção 3.2, uma matriz de adjacência completa pode ser mais passível de ser escolhida, por ser mais adequado e compacto manter linhas de uma matriz, a exemplo da apresentada na Figura 3.9, do que manter uma estrutura como a ilustrada na Figura 3.10, que pode facilmente se tornar um caso onde não há divisão das partes a serem processadas.

Com essa escolha, existe uma restrição de que as linhas completas devem ser mantidas em memória. No entanto, não há necessidade de carregar todos os casos ao mesmo tempo para processar as linhas. Para exemplificar, escolhe-se como tamanho para processamento, 3×3. Então, carregam-se 6 casos, para fazer a *combinação de todos com todos*, e sendo a primeira iteração, estes casos são os 3 primeiros, a exemplo dos casos 0, 1 e 2, duas vezes, uma vez em para iterar nas linhas e outra para as colunas. Após carregados, *calcula-se a distância entre todas as combinações*, e determina-se se estas estão próximas e coloca-se as adjacências em suas respectivas posições nas matrizes, como ilustrado na Figura 3.14. Depois, substitui-se os 3 casos que foram carregados em relação às colunas pelos próximos como, a exemplo, os casos carregados são 0, 1 e 2, para as linhas e 3, 4 e 5, para as colunas, como na Figura 3.15, repetindo até os últimos casos, observando que nas últimas iterações de cada bloco, pode haver menos casos para processar. No exemplo, a última iteração do primeiro bloco de linhas seria um bloco 3×1, a primeira do último bloco de linhas seria 1×3 e a última seria 1×1.

	j	0	1	2	3	4	5	6	7	8	9
i											
0		0	1	1	0	0	0	1	0	0	1
1		1	0	0	1	1	0	0	0	1	1
2		1	0	0	0	1	0	1	1	0	1
3		0	1	0	0	1	1	0	1	1	0
4		0	1	1	1	0	0	1	0	0	0
5		0	0	0	1	0	0	1	0	1	0
6		1	0	1	0	1	1	0	0	1	1
7		0	0	1	1	0	0	0	0	0	0
8		0	1	0	1	0	1	1	0	0	1
9		1	1	1	0	0	0	1	0	1	0

Figura 3.14: Circulado em vermelho destaca-se a restrição para linhas inteiras para cálculo de n_2 no algoritmo sem contadores e circulado em amarelo o bloco de processamento 3×3.

	j	0	1	2	3	4	5	6	7	8	9
i											
0		0	1	1	0	0	0	1	0	0	1
1		1	0	0	1	1	0	0	0	1	1
2		1	0	0	0	1	0	1	1	0	1
3		0	1	0	0	1	1	0	1	1	0
4		0	1	1	1	0	0	1	0	0	0
5		0	0	0	1	0	0	1	0	1	0
6		1	0	1	0	1	1	0	0	1	1
7		0	0	1	1	0	0	0	0	0	0
8		0	1	0	1	0	1	1	0	0	1
9		1	1	1	0	0	0	1	0	1	0

Figura 3.15: Segunda iteração do algoritmo em blocos.

Uma desvantagem para este algoritmo é que certas combinações de casos devem ser processadas duas vezes se a matriz é mantida parcialmente, como mostrado na Figura 3.16. A quantia pode ser reduzida conforme o tamanho dos blocos aumenta, mas só é eliminada quando o tamanho dos blocos é igual ao tamanho da matriz, ou seja, não há divisão nenhuma. Também os contadores devem ser mantidos para saber em que estado encontra-se o processamento, isto é, em que linha/coluna estaria processando se estivesse utilizando a matriz inteira em memória.

	j	0	1	2	3	4	5	6	7	8	9
i											
0		0	1	1	0	0	0	1	0	0	1
1		1	0	0	1	1	0	0	0	1	1
2		1	0	0	0	1	0	1	1	0	1
3		0	1	0	0	1	1	0	1	1	0
4		0	1	1	1	0	0	1	0	0	0
5		0	0	0	1	0	0	1	0	1	0
6		1	0	1	0	1	1	0	0	1	1
7		0	0	1	1	0	0	0	0	0	0
8		0	1	0	1	0	1	1	0	0	1
9		1	1	1	0	0	0	1	0	1	0

Figura 3.16: Circuladas em azul destacam-se as duplas que precisam ser processadas duas vezes.

Equivalentemente, para o caso com metade da matriz, utiliza-se a Figura 3.17 como exemplo. Para a solução do cálculo quadrático dos termos n_2 que utiliza contadores, não há necessidade de manter linhas inteiras em memória. Portanto, ao utilizar os mesmos tamanhos de

blocos, na primeira iteração sob a meia matriz de exemplo, tem-se o delimitado na Figura 3.18 mantido em memória, além de todos os n contadores. Para a segunda iteração, o bloco de linhas já é substituído de 0, 1 e 2 para 3, 4 e 5, resultando como na Figura 3.19.

	j	0	1	2	3	4	5	6	7	8	9
i											
0		-	-	-	-	-	-	-	-	-	-
1		1	-	-	-	-	-	-	-	-	-
2		1	0	-	-	-	-	-	-	-	-
3		0	1	0	-	-	-	-	-	-	-
4		0	1	1	1	-	-	-	-	-	-
5		0	0	0	1	0	-	-	-	-	-
6		1	0	1	0	1	1	-	-	-	-
7		0	0	1	1	0	0	0	-	-	-
8		0	1	0	1	0	1	1	0	-	-
9		1	1	1	0	0	0	1	0	1	-

Figura 3.17: Meia matriz de exemplo.

	j	0	1	2	3	4	5	6	7	8	9
i											
0		-	-	-	-	-	-	-	-	-	-
1		1	-	-	-	-	-	-	-	-	-
2		1	0	-	-	-	-	-	-	-	-
3		0	1	0	-	-	-	-	-	-	-
4		0	1	1	1	-	-	-	-	-	-
5		0	0	0	1	0	-	-	-	-	-
6		1	0	1	0	1	1	-	-	-	-
7		0	0	1	1	0	0	0	-	-	-
8		0	1	0	1	0	1	1	0	-	-
9		1	1	1	0	0	0	1	0	1	-

Figura 3.18: Primeira iteração no algoritmo com meia matriz.

	j	0	1	2	3	4	5	6	7	8	9
i											
0		-	-	-	-	-	-	-	-	-	-
1		1	-	-	-	-	-	-	-	-	-
2		1	0	-	-	-	-	-	-	-	-
3		0	1	0	-	-	-	-	-	-	-
4		0	1	1	1	-	-	-	-	-	-
5		0	0	0	1	0	-	-	-	-	-
6		1	0	1	0	1	1	-	-	-	-
7		0	0	1	1	0	0	0	-	-	-
8		0	1	0	1	0	1	1	0	-	-
9		1	1	1	0	0	0	1	0	1	-

Figura 3.19: Segunda iteração no algoritmo com meia matriz.

A utilização de contadores resolve o problema de manter linhas completas em memória, tanto para a primeira abordagem de divisão em blocos quando para a segunda. E a questão do múltiplo processamento de duplas, visto na divisão de blocos com matriz completa, é resolvida pela divisão com matriz parcial.

Observa-se que podem haver diferenças nos resultados das estatísticas calculadas das permutações de Monte Carlo conforme escolha do método de divisão em blocos, associado com o método de amostragem utilizado, dado a diferença da ordenação/quantia dos dados presentes nas matrizes de adjacência. Nota-se também que, como nas soluções para o cálculo de n_{2t} e n_{2s}, as divisões em blocos são aplicadas da mesma maneira para a matriz temporal e espacial. Outra consequência da divisão em blocos, é a necessidade de manter contadores parciais para a estatística observada, n_s, n_t e cada estatística a ser calculada pelas permutações de Monte Carlo, se forem utilizadas. Nelas permuta-se um bloco, calcula-se a estatística para o bloco e armazena em um contador, realiza-se outra permutação e armazena-se no próximo contador, e assim sucessivamente.

Para o algoritmo de blocos com contadores e matrizes parciais, um Fluxograma é dado pela Figura 3.20, junto com o Algoritmo 7.

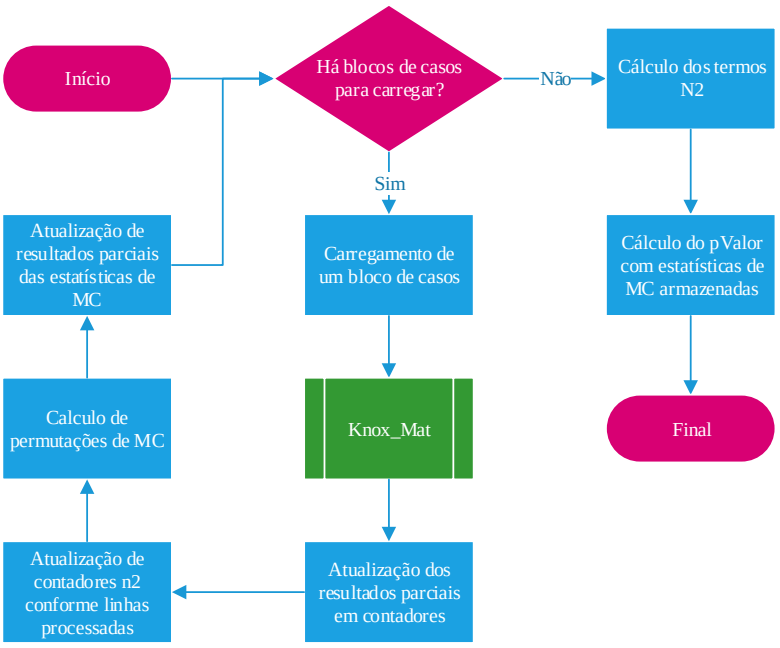

Figura 3.20: Fluxograma para o algoritmo de blocos com contadores e matrizes parciais, onde "Knox_Mat" é o cálculo da estatística de Knox.

Algoritmo 7: PSEUDOCÓDIGO PARA O ALGORITMO DE BLOCOS COM CONTADORES E MATRIZES PARCIAIS

```
1 início
2     para cada bloco de casos faça
3         Cálculo das matrizes de adjacência para o bloco de casos;
4         Atualização dos contadores parciais (Estatística observada, n_s e n_t);
5         Atualização dos contadores dos termos n2;
6         Cálculo das permutações de Monte Carlo;
7         Atualização dos contadores parciais das estatísticas de MC;
8     fim
9 fim
```

3.5 Algoritmo Incremental

Para implementar a variação incremental do método de Knox, isto é, para implementar o iKnox, utiliza-se o Algoritmo 1 para calcular a Estatística do método para cada par de parâmetros críticos, alterando-se apenas a determinação de proximidade temporal para os Algoritmos 2 e 3.

Uma dupla de casos é considerada próxima no tempo se a distância é menor que um limiar. Porém, na abordagem incremental, casos são considerados próximos no tempo se a distância entre eles estiver contida em um intervalo ou for igual à um limiar. Por exemplo, se o intervalo em questão for 5 dias, na abordagem original a proximidade seria determinada por $d_{ij}^{T}(i,j) < Crit_T$, enquanto na incremental seria $d_{ij}^{T}(i,j) = Crit_T$, isto é, os casos i e j seriam considerados adjacentes apenas se a distância temporal entre eles fosse exatamente 5 dias. Se, na incremental, o intervalo for entre 2 e 5 dias, a proximidade é determinada por $Crit_{T1} \leqslant d_{ij}^{T}(i,j) \leqslant Crit_{T2}$, isto é, são próximos i e j se a distância entre eles pertencer ao intervalo $[2,5]$.

Como discutido na Seção 2.2 e exemplificado na Seção 2.2.1, essa abordagem requer que a estatística seja calculada diversas vezes, e o algoritmo geral, com as alterações para determinação de proximidade, pode ser encapsulado em outro algoritmo como no Fluxograma da Figura 3.21 e no seu respectivo Algoritmo 8.

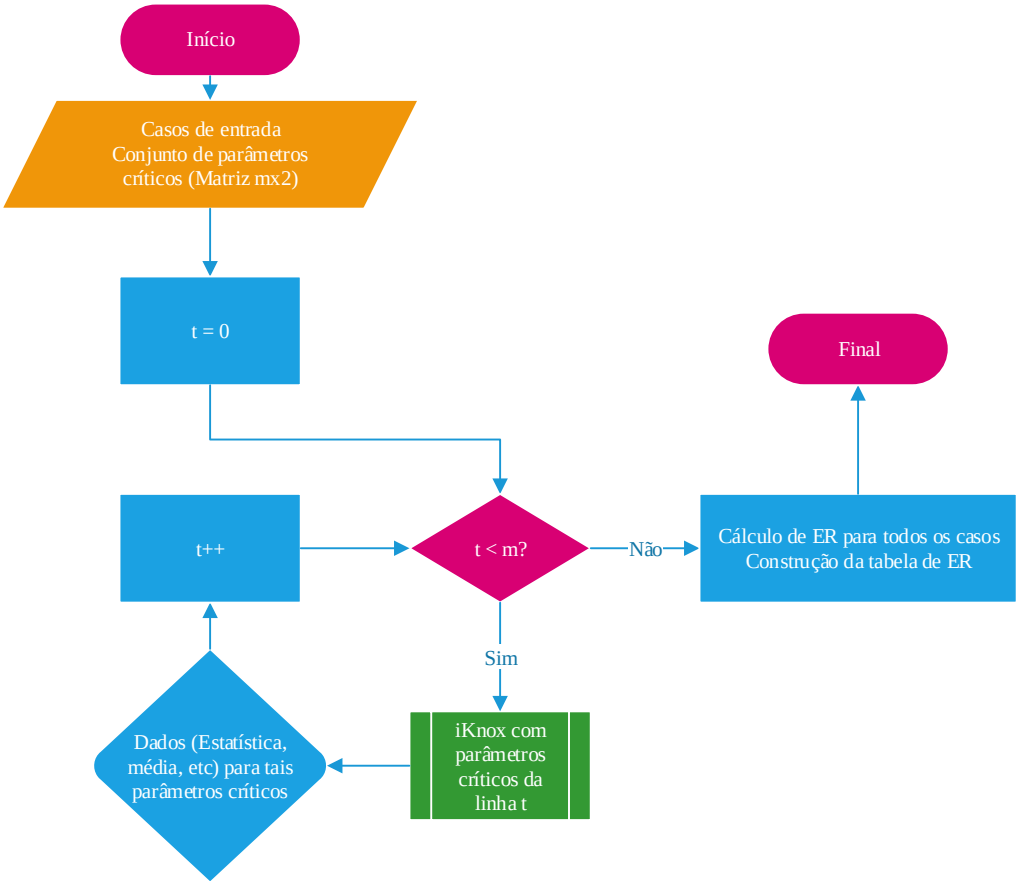

Figura 3.21: Fluxograma geral do algoritmo knox incremental.

Algoritmo 8: ALGORITMO GERAL DE IKNOX

início
    \\Array de n elementos indexado de 0 até $n-1$
    \\crit é uma matriz $m \times 2$, onde $m[][0]$ é $Crit_S$, e $m[][1]$ é $Crit_T$
    $t = 0$
    para t *de* 0 *até* $m-1$ **faça**
        Knox-Geral$(crit[t][0], crit[t][1], Casos - Entrada)$
        \\Algoritmo geral de Knox
    fim
    \\Construção de Tabelas *Elevated Risk*
fim

Outro detalhe a se notar está relacionado a característica inerente à esse método na execução de várias testes sequenciais, alterando apenas os parâmetros de entrada, como mostrado na Tabela 2.10 da Seção 2.2.1. Esse aspecto associado ao fato de que se utilizar os mesmos parâmetros de entrada gerariam-se as mesmas matrizes e resultados parciais, leia-se n_s e n_t, o encapsulamento de exemplo apresenta diversos cálculos redundantes, repetindo três vezes o cálculo da mesma matriz durante as várias iterações do exemplo citado, como o cálculo de matriz espacial com $\delta_S = 0,5$. Para remover essa redundância ter-se-ia que armazená-las para recuperação futura, que pode ser subsequente, no caso das espaciais, ou não, como no caso das temporais. O problema com essa solução está na velocidade de acesso do armazenamento secundário, que torna-se necessário para n grande, pelas matrizes ocuparem muito da memória primária, ou um *overhead* muito grande para n pequeno, pois o tempo de cálculo pode-se tornar muito próximo do tempo de recuperação dos valores armazenados.

3.6 Exemplo

Para ilustrar a organização dos dados em memória bem como a aplicação dos algoritmos sobre eles, utilizam-se os dados já calculados dos exemplos anteriores, como nas seções 2.1.2 e 2.2.1. Primeiramente, calcula-se a estatística de Knox e a média do teste; depois calcula-se ora a aproximação $p - Valor_{Poisson}$ ora a simulação $p - Valor_{MC}$; então mostra-se detalhes para os cálculos envolvidos em um exemplo de iKnox.

3.6.1 Estatística de Knox

Antes de calcular a estatística do método, alocam-se as estruturas necessárias para tal. No caso do código de Tango (2010), matrizes quadradas $n \times n$, como nas Tabelas (3.1) e (3.2); no caso de Höhle et al. (2017), vetores, como na Tabela (3.3) e na solução proposta ora matrizes triangulares sem a diagonal principal, como nas Tabelas (3.4) e (3.5), ora vetores. A utilização de matrizes parciais na solução apresentada é somente obrigatória caso opte-se pelo cálculo da variância por n_{2s} e n_{2t}. Observa-se que para este último, pode-se precisar da alocação de contadores, caso seja essa a estratégia selecionada. Como os dados são iguais aos exemplos supracitados, pode-se compará-los com a Tabela (2.5). Os casos nas matrizes foram indexados como na linguagem C de modo que o caso 1 seja acessado como 0.

Tabela 3.1: Matriz completa com adjacências espaciais, onde a primeira linha ou coluna são os casos.

	0	1	2	3	4
0	0	0	0	0	1
1	0	0	0	0	0
2	0	0	0	1	0
3	0	0	1	0	0
4	1	0	0	0	0

Tabela 3.2: Matriz completa com adjacências temporais, onde a primeira linha ou coluna são os casos.

	1	2	3	4	5
1	0	1	1	0	0
2	1	0	1	0	0
3	1	1	0	1	0
4	0	0	1	0	1
5	0	0	0	1	0

Tabela 3.3: Vetores com adjacências espaciais e temporais.

Dupla	1	2	3	4	5	6	7	8	9	10
Espacial	0	0	0	1	0	0	0	1	0	0
Temporal	1	1	0	0	1	0	0	1	0	1

Tabela 3.4: Matriz parcial com adjacências espaciais, onde a primeira linha ou coluna são os casos e "$-$" é uma posição inexistente ou não alocada.

	0	1	2	3	4
0	-	-	-	-	-
1	0	-	-	-	-
2	0	0	-	-	-
3	0	0	1	-	-
4	1	0	0	0	-

Tabela 3.5: Matriz parcial com adjacências temporais, onde a primeira linha ou coluna são os casos e "$-$" é uma posição inexistente ou não alocada.

	0	1	2	3	4
0	-	-	-	-	-
1	1	-	-	-	-
2	1	1	-	-	-
3	0	0	1	-	-
4	0	0	0	1	-

Observa-se que as duas últimas podem ser tanto triangulares inferiores quanto superiores, a única adaptação necessária seria a ordem de consideração das duplas, de modo que a dupla 1 da Tabela (2.5) passaria a ser indexada como os casos $(2, 1)$ ao invés de $(1, 2)$ em um contexto $(Linha, Coluna)$.

Como no exemplo anterior, com as adjacências pode-se obter n_{st}, n_s, n_t e, consequentemente, $E(n_{st})$ e $p - Valor_{Poisson}$. Para obter os três primeiros de maneira algorítmica, basta-se iterar sobre as posições das matrizes de adjacência. Assim, a primeira iteração do laço se aplicado sobre as matrizes triangulares inferiores interagiria com os valores das variáveis como $n_s+ = Mat_S(1, 0)$, $n_t+ = Mat_T(1, 0)$ e $n_{st}+ = Mat_S(1, 0)\ AND\ Mat_T(1, 0)$. Similarmente, se aplicado sobre os vetores, teriam-se as operações $n_s+ = Vet_S(0)$, $n_t+ = Vet_T(0)$ e $n_{st}+ = Vet_S(0)\ AND\ Vet_T(0)$.

3.6.2 Variância

Para o cálculo de Var sem MC necessitam-se dos valores de n_{2s} e n_{2t}, para tanto, a organização dos dados em vetores não pode ser utilizada, restringindo a organização dos dados em matrizes completas, como em Tango (2010), ou parciais, como na solução proposta. Logo em seguida, apenas a última é considerada, mostrando as abordagens de "contadores"e de "linhas",

com as devidas adaptações devido ao uso de matrizes triangulares, como destacado na Figura 3.10.

Para a abordagem dos contadores, como discutido na Seção 3.2, precisam-se de no mínimo um contador por linha, isto é, n contadores, como na Tabela (3.6), ou $n \cdot 2$ contadores, como na Tabela (3.7). Observa-se que ambas as Tabelas contem adjacências de espaço para simplificar a exemplificação, sendo que as mesmas operações devem ser aplicadas sobre as adjacências temporais.

Tabela 3.6: Matriz com adjacências espaciais e um contador.

	0	1	2	3	4	x
0	-	-	-	-	-	1
1	0	-	-	-	-	0
2	0	0	-	-	-	1
3	0	0	1	-	-	1
4	1	0	0	0	-	1

Tabela 3.7: Matriz com adjacências espaciais e dois contadores.

	0	1	2	3	4	x1
0	-	-	-	-	-	0
1	0	-	-	-	-	0
2	0	0	-	-	-	0
3	0	0	1	-	-	1
4	1	0	0	0	-	1
x2	1	0	1	0	0	

Com as matrizes, executa-se o algoritmo para cálculo dos termos. Assim, realiza-se a contagem de 1s conforme a abordagem selecionada, resultando em uma das Tabelas acima. Após a contagem, realiza-se um laço passando por cada uma das n posições do(s) contador(es), incrementando o resultado final, de modo que a primeira iteração para um contador ficaria como $n_{2s} += x(0) \cdot (x(0) - 1)$ e com dois contadores ficaria como $n_{2s} += (x1(0) + x2(0)) \cdot ((x1(0) + x2(0)) - 1)$, executando o equivalente para a matriz temporal. A abordagem de linhas é semelhante à da Tabela (3.6), mas ao invés de armazenar a quantia de 1s para cada linha, o incremento é realizado logo após a contagem.

3.6.3 Simulação de Monte Carlo

Para as simulações de MC, precisa-se realizar a amostragem da matriz temporal de modo que a Tabela (3.5) poderia ser alterada para (3.8). Observa-se que a amostragem de exemplo tem o mesmo resultado da primeira simulação da seção (2.1.2) todas as explicitadas no Apêndice A, no formato vetorial.

Tabela 3.8: Matriz parcial com adjacências temporais amostrada.

	0	1	2	3	4
0	-	-	-	-	-
1	1	-	-	-	-
2	1	0	-	-	-
3	1	1	0	-	-
4	0	0	1	0	-

Dada a primeira amostragem temporal acima, tem-se que o valor de n_{st} para esta simulação é 0, pois nenhuma posição possui 1 em ambas as matrizes simultaneamente, diferente dos dados observados, onde isso ocorre uma vez. Com a estatística do método calculada, pode-se utilizá-la para o cálculo do Rank, da média e variância das simulações.

Observa-se que Tango (2010) faz a amostragem diretamente sobre os dados de entrada, tornando necessário o cálculo de todas as distâncias entre todas as combinações novamente, bem como as adjacências, impactando severamente no desempenho das simulações de MC. No entanto, como calculam-se as distâncias e adjacências de todas as combinações possíveis para a estatística observada, pode-se realizar a amostragem sobre as matrizes ou vetores de adjacência diretamente, como no caso de Meyer (2017).

3.6.4 Organização em blocos

Como para casos onde n é grande há um consumo de memória elevado, pode-se adotar estratégias que visam diminuir esse problema, ao custo de certas limitações*trade-off*s, como a técnica de paginação, que é invisível ao programa, mas pode ter desempenho reduzido devido a lentidão do armazenamento em discos rígidos ou a organização em blocos, onde há limitações nas escolhas de algoritmos para certas etapas e na aleatoriedade das amostragens para MC.

Na etapa da estatística observada, há apenas uma pequena restrição na divisão por blocos, que as duplas estejam alinhadas entre espacial e temporal, ou seja, que o índice i dos dados

de entrada contenha a posição espacial e temporal para o mesmo caso ou que o índice da matriz\vetor de adjacências possui a mesma dupla para espaço e tempo, caso a computação das distâncias seja feita como uma preparação dos dados.

A etapa de n_2, no entanto, apresenta restrições independente do método selecionado. No método por contadores, os blocos não possuem restrições quanto ao tamanho, mas precisam de informações de linha e coluna das duplas para que os contadores corretos sejam atualizados. No método por linhas, não há necessidade de informações posicionais adicionais, mas há restrições quanto ao tamanho dos blocos, que devem ser de $n \times k$, onde n é referente às linhas e k é um tamanho genérico. Essa restrição é melhor aplicada para matrizes completas, pois em parciais, a "linha" é refletida em uma coluna, vide Figura 3.10. No primeiro caso, as informações associadas podem ser simplesmente um *offset* para linha e um para coluna, já que a posição exata pode ser descoberta com este último e o deslocamento interno do bloco.

Como etapa de MC é a re-execução da etapa de n_{st}, algo similar pode ser dito para ela. Além da restrição de posições, há limitações quanto a amostragem. Com as matrizes completamente em memória, pode-se trocar todas as posições com todas as outras, mas quando se tem apenas parte delas, os blocos, a troca aleatória fica restrita dentro do próprio bloco, reduzindo as possibilidades de amostras únicas.

Para exemplificar, realiza-se a divisão em blocos de tamanho 2×2, como na Tabela (3.9) nas matrizes e na Tabela (3.10) nos vetores. Diferentemente do sugerido na Tabela, durante a execução do algoritmo, apenas um bloco é mantido em memória por vez, passando o mesmo por todas as etapas antes que seja substituído pelo próximo.

Tabela 3.9: Divisão da matriz parcial espacial em blocos 2×2, onde cada cor é referente a um bloco, e $+$ para a diagonal principal.

	0	1	2	3	4
0	+	-	-	-	-
1	0	+	-	-	-
2	0	0	+	-	-
3	0	0	1	+	-
4	1	0	0	0	+

Tabela 3.10: Vetor com adjacências espaciais em blocos de tamanho 4, onde cada cor é referente a um bloco.

Espacial	0	0	0	1	0	0	0	1	0	0

Uma maneira de realizar a criação dos blocos é percorrendo a lista de casos com dois ponteiros, um equivalente à linha da matriz de adjacência e outro para a coluna, de modo que para a primeira iteração de exemplo, teria-se o caso 0 apontado pelo ponteiro da linha e coluna. Assim, o primeiro bloco seria formado pelas duplas $(0,0)$, $(0,1)$, $(1,0)$ e $(1,1)$, onde apenas $(1,0)$ seria considerada, já que é a única que está abaixo da diagonal principal, com *offset* 0 na linha e 0 na coluna; o segundo bloco seria formado por $(2,0)$, $(2,1)$, $(3,0)$ e $(3,1)$, com todas consideradas e offset 2 para linha e 0 para coluna. Para descobrir qual o caso exato com o uso do *offset*, testa-se com o terceiro caso do segundo bloco, $(3,0)$, em que sua posição interna no bloco é $(1,0)$, tal que, usando (3.3), Posição Externa$(3,0)$ $=$ Posição Interna$(1,0)$ $+$ Offset$(2,0)$. O *offset* pode ser entendido como a posição absoluta do primeiro caso no bloco atual.

$$\text{Posição Externa}(x,y) = \text{Posição Interna}(x1,y1) + \text{Offset}(x2,y2) \tag{3.3}$$

Para ilustrar o cálculo de todas as métricas discutidas até então, realizam-se dois exemplos, o primeiro com n_{st}, $E(n_{st})$ $Var(n_{st})$ e MC, usando, consequentemente, matrizes e o segundo com as mesmas métricas, exceto por $Var(n_{st})$, podendo simplificar a execução somente com vetores.

No primeiro caso, inicia-se alocando duas matrizes com o tamanho desejado, isto é, 2×2; inicializam-se os identificadores de descolamento (*offset*) para linhas ou colunas em 0 e calculam-se critérios para determinar os blocos, então obtêm-se as distâncias e determinam-se as adjacências entre os casos determinados pelos índices externos atuais, conforme (3.3), em cada bloco, em seguida passando o bloco pelos algoritmos de n_{st}, n_2 e MC. A não ser que ambas as dimensões do bloco sejam divisores de n, haverão blocos parciais, de modo que pode-se classificá-los em categorias. Estas são "linha e colunas inteiras", "linhas parciais e colunas inteiras", "linhas parciais e colunas inteiras"e "linhas e colunas parciais". O tratamento desses blocos nas três últimas categorias é pertinente para não realizar cálculos desnecessários, é pode ser resolvido em duas maneiras. A primeira é realizar apenas um controle de índices, onde posições acima ou abaixo da diagonal principal, conforme escolha de matriz triangular, não fatoram no cálculo e, consequentemente, no resultado final. A segunda é semelhante, mas preenche as posições ignoradas com zero, para que quando esses passem pelas partes do algoritmo que não precisam de índices associados não alterem o resultado. Observa-se que na etapa de amostra-

gem do MC o controle de índices é imprescindível para que as posições zeradas ou ignoradas não sejam consideradas como opções de troca.

Ilustrando o presente caso, considera-se o primeiro bloco que, como supracitado, é formado pelas duplas $(0, 0)$, $(0, 1)$, $(1, 0)$ e $(1, 1)$. Dentro da execução do algoritmo, apenas a dupla $(1, 0)$ terá suas distâncias calculadas, ao ponto que será feita a limiarização. Ao fim dessa etapa, tem-se um bloco para o espaço e outro para o tempo com os devidos valores de adjacência colocados, para que possam atualizar a contagem de n_{st}, n_s e n_t, como na Tabela (3.11). Nota-se que pode-se combinar as duas etapas em uma só, como no Fluxograma da Figura 3.2, com elas juntas, *versus* o Fluxograma da Figura 3.3, com elas separadas. Em seguida, os contadores de n_{2s} e n_{2t} são atualizados, alterando os contadores como na Tabela (3.12). Por fim, os blocos pela etapa de MC, atualizando os contadores de todas as simulações, como na Tabela (3.13). Nota-se que nesse exemplo é possível notar a restrição de blocos pequenos para quantidade grande de simulações de MC, onde há apenas uma possibilidade de amostragem do bloco temporal aqui, com um elemento e uma posição mas precisa-se de pelo menos dez, considerando a observada e nove de simulações.

Tabela 3.11: Primeiro bloco espacial e primeiro temporal, e o estado dos contadores de n_{st}, n_s e n_t.

Espacial		Temporal		n_{st}	n_s	n_t
+	-	+	-	0	0	1
0	+	1	+			

Tabela 3.12: Estado dos vetores de contadores de n_{2s} e n_{2t}.

	Espacial		Temporal	
	x1	x2	x1	x2
0	0	0	0	1
1	0	0	1	0
2	0	0	0	0
3	0	0	0	0
4	0	0	0	0

Tabela 3.13: Estado dos contadores das simulações de MC ao fim da passagem do primeiro bloco espacial e do primeiro temporal.

MC	n_{st}
1	0
2	0
3	0
4	0
5	0
6	0
7	0
8	0
9	0

Esse bloco é substituído pelo segundo. Então realiza-se o cálculo das distâncias, das adjacências e a atualização de n_{st}, n_s e n_t, resultando na Tabela (3.14); depois realiza-se a atualização dos contadores de n_2, resultando na Tabela (3.15); finalmente, realiza-se a atualização dos contadores das simulações de MC, resultando na mesma Tabela (3.13), já que o bloco espacial possui apenas zeros. Nota-se que no segundo bloco temporal, já existem $4!$ possibilidades de amostragem, sendo superior às dez necessárias.

Tabela 3.14: Segundo bloco espacial e segundo temporal, e o estado dos contadores de n_{st}, n_s e n_t.

Espacial		Temporal		n_{st}	n_s	n_t
0	0	1	1	0	0	3
0	0	0	0			

Tabela 3.15: Estado dos vetores de contadores n_{2s} e n_{2t} após atualização do segundo bloco.

	Espacial		Temporal	
	x1	x2	x1	x2
0	0	0	0	2
1	0	0	1	1
2	0	0	2	0
3	0	0	0	0
4	0	0	0	0

No segundo caso, aloca-se dois vetores do tamanho desejado, como 4 por exemplo, iniciam-se dois contadores para controle das combinações então calculam-se as distâncias e adjacências para o bloco atual, o passando pelo restante dos algoritmos logo em seguida. Os contadores não

impactam em quais duplas serão colocadas dentro do bloco, apenas como critérios de parada para os laços que os criam. Eles podem ser utilizados de modo a simular uma matriz triangular inferior ou superior dependendo qual contador, coluna ou linha, respectivamente, é usado no *loop* mais interno e qual, coluna ou linha, respectivamente, é utilizado no mais externo. Na inferior, o interno inicia em zero e vai até $i - 1$ inclusive e o externo de 1 até n exclusive; na superior, o interno vai de $i + 1$ inclusive à n exclusive. Observa-se que nessa abordagem, há uma maior economia de espaço, já que os blocos não apresentam valores insignificantes, exceto talvez pelo último, se o tamanho dos blocos não for um divisor de n.

Para ilustrar as diferenças entre esse caso e os blocos por matriz, comparam-se as Tabelas (3.16) e (3.11) para tais escolhas, respectivamente. As principais vantagens do atual são a economia de espaço, já que os quatro primeiros casos estão no bloco, contra apenas um no anterior e menor quantia de blocos a ser processada, já que há menos blocos com casos insignificantes; enquanto a principal desvantagem é a impossibilidade de calcular $Var(n_{st})$ com a expressão (2.6) sem a associação de estruturas para controle de índices e posições das duplas dentro dos blocos.

Tabela 3.16: Primeiro bloco espacial e primeiro temporal, e o estado dos contadores de n_{st}, n_s e n_t.

Espacial	0	0	0	0	n_{st}	n_s	n_t
Temporal	1	1	1	0	0	0	3

Capítulo 4

Resultados

4.1 Clusterização espaço-temporal de Dengue

A dengue é uma doença viral transmitida entre humanos por mosquitos fêmeas infectados das espécies *aedes aegypti* e *aedes albopictus*, onde a primeira origina a grande parte dos casos. Os mosquitos, que também transmitem chikungunya, zika e febre amarela, tornam-se portadores da doença ao entrarem em contato com seres humanos infectados. Após o período de incubação do vírus por 4 a 10 dias, um mosquito pode transmiti-lo pelo resto de sua vida, que é de cerca de 45 dias (WHO., 2016).

Nos humanos, a incubação possui um período de 4 a 10 dias sucedidos à picada de um mosquito infectado. A infecção pelo vírus então causa febre alta, em torno de 40° C, e pelo menos dois sintomas dentre dor de cabeça intensa, dor atrás dos olhos, dores musculares e articulares, náuseas, vômitos, erupções na pele, que duram de 2 a 7 dias (WHO., 2018a).

Enquanto a dengue dificilmente causa morte, repetida exposição à diferentes sorotipos da dengue, denominados DEN-1, DEN-2, DEN-3 e DEN-4, pode causar uma complicação alcunhada de dengue severa, que pode ser fatal no período de 24 a 48 horas do estágio crítico. Dengue não possui tratamento específico (WHO., 2018a).

Apesar de erros de classificação e da substancial falta de notificação em órgãos de saúde e para a Organização Mundial da Saúde (WHO), o número de casos reportados aumentou de 2,2 milhões em 2010 para 3,2 milhões em 2015. Com essas considerações, estima-se, que os casos sintomáticos chegam a faixa de 50 a 100 milhões nos anos recentes, predominantemente na Asia, sucedido pela América Latina e a Africa (WHO., 2016).

Dentre os fatores que influenciam fortemente a difusão da dengue, listam-se a precipitação,

a temperatura e a urbanização rápida não planejada, especialmente em regiões sem o tratamento adequado de água e lixo, gerando criadouros adequados para o aumento na proliferação do mosquito. Até o momento, embora diversas estratégias existentes de controle Rather (2017), o principal método para controlar ou prevenir a transmissão do vírus da dengue é combater os vetores (WHO., 2018b).

A realização de estudos a partir de análises da relação entre mosquitos e seres humanos, considerando o tempo e o lugar, pode contribuir para o melhor entendimento da dinâmica espacial da dengue e também para a execução de ações mais específicas e eficazes ao controle e combate vetorial (OLIVERIA M. A.; RIBEIRO, 2013) . Os métodos de análise geoespacial podem fornecer informações sobre áreas de risco, onde as medidas de controle e combate podem ser focalizadas reduzindo a transmissão do vírus em comparação com intervenções aleatórias ou essencialmente empíricas (VICENTI-GONZALEZ, 2017).

Em se considerando a variação espaço-temporal dos casos passa-se a conceber a não-aleatoriedade na distribuição da doença em que, entre eventos próximos no tempo, há um quantidade significativa de eventos que também estão próximos no espaço (JACQUEZ, 2008). Mais especificamente, a existência de *clusters* indicam um número incomum de casos em uma população de uma região em um certo período de tempo e que pode ser identificado, visualizado e explorado utilizando métodos de análise espacial (MCLAFFERTY, 2015).

A literatura sobre métodos estatísticos para a clusterização de doenças é significativa, sendo o texto de Tango (2010) um exemplo significativo. Esse autor discute a clusterização espaço-temporal e vários métodos para tratá-los entre eles Knox, Mantel, Max Baker, Jacquez, Diggle, Dulldoff e Hjalmars. Desses, o método de Knox é o mais utilizado e foi historicamente empregado para explorar a clusterização de Leucemia (KNOX, 1964).

No método de Knox, cada caso da doença a ser estudada dentro de um período é associado ao tempo de ocorrência e a localização geográfica. Então, combinam-se os casos em duplas e calculam-se as distâncias temporais e espaciais entre eles, classificando-as em quatro categorias. A primeira com pares próximos no espaço e no tempo, a segunda e a terceira com pares próximos apenas no espaço ou apenas no tempo e a última com pares distantes. Uma grande concentração de duplas na primeira categoria indica uma interação espaço-temporal. A vantagem em utilizar este método é que ele é intuitivo, elegante e simples. Embora receba crí-

ticas quanto a arbitrariedade envolvida na seleção do tempo e distância críticos, bem como das considerações sobre as mudanças na população envolvida, essas questões podem ser melhor analisadas aplicando-se Monte Carlo (GEAR, 2006).

Nesse contexto, foram realizados testes com o método de iKnox para dois arquivos de dados, o primeiro para Cascavel com 1.801 casos e o segundo para o Rio de Janeiro capital, com 225.704 casos. Destaca-se que a execução das 230 combinações de parâmetros críticos de Cascavel não ultrapassou 5 minutos, enquanto cada um dos 90 testes do Rio de Janeiro levou cerca de 167 minutos.

4.1.1 Caso Cascavel

A Figura 4.1 mostra os pontos de ocorrência distribuídos espacialmente, e a Figura 4.2 mostra a distribuição das estatísticas em cada testes iKnox executado. Nota-se que o máximo global de cada distribuição espacial ao longo do tempo situa-se quando $\delta_T = 0$, ou seja, o maior número de casos clusterizados em qualquer espaço testado ocorreram no mesmo dia.

Figura 4.1: Distribuição espacial dos casos de Cascavel.

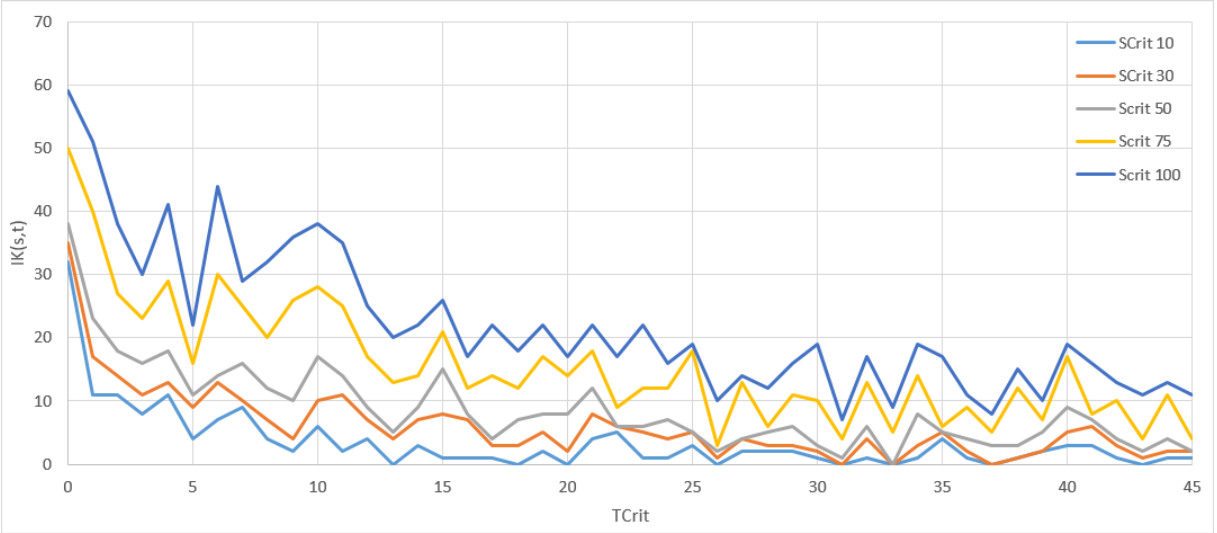

Figura 4.2: Distribuição da estatística de iKnox onde o eixo Y é referente aos valores da estatística, o eixo X do tempo crítico e cada linha é um espaço crítico.

As Tabelas (4.1) e (4.2) apresentam os dados utilizados para a construção do gráfico da Figura 4.2.

Tabela 4.1: Primeira parte dos valores da estatística de iKnox nos testes realizados com dados de Cascavel.

TCrit.	SCrit. 10	SCrit. 30	SCrit. 50	SCrit. 75	SCrit. 100
0	32	35	38	50	59
1	11	17	23	40	51
2	11	14	18	27	38
3	8	11	16	23	30
4	11	13	18	29	41
5	4	9	11	16	22
6	7	13	14	30	44
7	9	10	16	25	29
8	4	7	12	20	32
9	2	4	10	26	36
10	6	10	17	28	38
11	2	11	14	25	35
12	4	7	9	17	25
13	0	4	5	13	20
14	3	7	9	14	22
15	1	8	15	21	26
16	1	7	8	12	17
17	1	3	4	14	22
18	0	3	7	12	18
19	2	5	8	17	22
20	0	2	8	14	17
21	4	8	12	18	22
22	5	6	6	9	17

Tabela 4.2: Segunda parte dos valores da estatística de iKnox.

TCrit.	SCrit. 10	SCrit. 30	SCrit. 50	SCrit. 75	SCrit. 100
23	1	5	6	12	22
24	1	4	7	12	16
25	3	5	5	18	19
26	0	1	2	3	10
27	2	4	4	13	14
28	2	3	5	6	12
29	2	3	6	11	16
30	1	2	3	10	19
31	0	0	1	4	7
32	1	4	6	13	17
33	0	0	0	5	9
34	1	3	8	14	19
35	4	5	5	6	17
36	1	2	4	9	11
37	0	0	3	5	8
38	1	1	3	12	15
39	2	2	5	7	10
40	3	5	9	17	19
41	3	6	7	8	16
42	1	3	4	10	13
43	0	1	2	4	11
44	1	2	4	11	13
45	1	2	2	4	11

A Figura 4.3 apresenta os valores normalizados das estatísticas apresentadas anteriormente distribuídas em um gráfico. As Tabelas (4.3) e (4.4), por sua vez, apresentam os dados utilizados para esse gráfico.

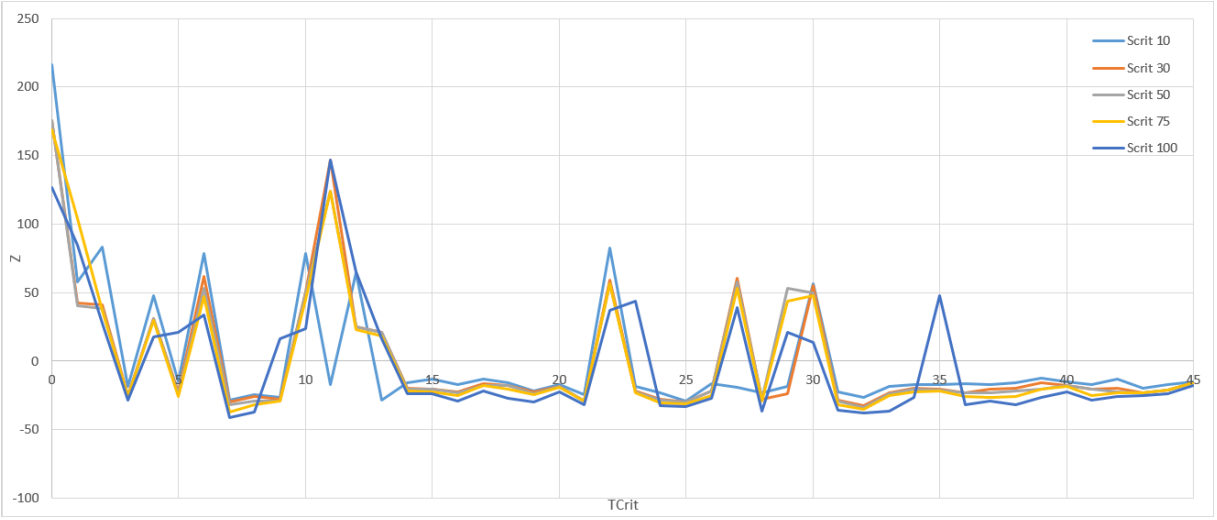

Figura 4.3: Distribuição dos valores de Z nos testes em Cascavel para a estatística de iKnox onde o eixo Y é referente aos valores de Z, o eixo X do tempo crítico e cada linha é um espaço crítico.

Tabela 4.3: Primeira parte dos valores de Z nos testes realizados com dados de Cascavel.

TCrit.	SCrit. 10	SCrit. 30	SCrit. 50	SCrit. 75	SCrit. 100
0	216,5895	175,4974	175,4974	168,6776	126,4873
1	57,84984	42,22933	40,28621	103,696	84,46229
2	83,11581	40,92865	38,51637	36,95094	27,79726
3	-18,646	-25,1709	-25,5896	-25,5896	-28,2631
4	47,77458	31,10419	31,10419	29,97333	17,68348
5	-15,1963	-22,5285	-23,5716	-26,0617	20,82295
6	78,83016	62,13812	53,41176	47,13832	33,53648
7	-28,4197	-29,4767	-31,4724	-36,9279	-41,337
8	-24,5246	-25,5194	-29,4387	-31,971	-37,2409
9	-26,6263	-27,5981	-29,0119	-29,2732	16,14388
10	78,43482	50,55687	49,58433	45,22126	23,94371
11	-16,8057	147,0269	123,9429	123,9429	146,6379
12	64,27477	25,14795	25,14795	23,3561	64,92685
13	-28,5543	21,20643	21,20643	18,10596	16,49331
14	-15,6516	-19,7709	-20,3342	-21,5081	-23,5278
15	-13,2076	-20,3705	-20,3705	-22,22	-23,8229
16	-17,2818	-22,4495	-23,4234	-25,2625	-29,0104
17	-13,1045	-16,4243	-17,4358	-17,9258	-21,9626
18	-15,7073	-17,7478	-18,0657	-20,6025	-27,2402
19	-21,8946	-22,3079	-23,7897	-24,5009	-29,7709
20	-16,7892	-18,1959	-18,1959	-19,0472	-22,174
21	-24,5523	-28,8691	-29,2251	-29,2626	-31,8527
22	82,5447	59,17633	56,31048	56,31048	36,76905

Tabela 4.4: Segunda parte dos valores de Z.

TCrit.	SCrit. 10	SCrit. 30	SCrit. 50	SCrit. 75	SCrit. 100
23	-18,6122	-21,9887	-22,4093	-23,0469	43,63714
24	-23,0603	-28,0624	-28,3586	-30,5006	-32,1694
25	-28,8697	-29,5485	-29,7238	-30,8082	-32,7855
26	-16,6071	-21,4793	-21,4793	-25,0281	-27,1183
27	-19,2316	60,38651	58,03992	52,85787	39,2076
28	-22,9224	-28,0958	-28,0958	-28,964	-36,3266
29	-18,4564	-23,873	52,98269	43,69381	20,72899
30	56,56044	54,8994	49,86026	47,57015	13,56425
31	-22,3792	-28,2911	-29,2265	-31,6465	-35,8517
32	-26,3428	-32,1461	-34,1001	-34,9906	-37,9504
33	-18,2638	-22,8941	-24,0415	-24,8381	-36,15
34	-16,7843	-19,8997	-20,3198	-22,598	-26,3524
35	-17,3948	-20,2644	-20,8989	-21,8152	47,52539
36	-16,1606	-22,7941	-23,089	-25,9323	-31,838
37	-17,0714	-20,1329	-22,7637	-26,1824	-29,3079
38	-15,4514	-19,5394	-21,9268	-25,5174	-31,4984
39	-12,4338	-15,8403	-20,2742	-20,6281	-26,6723
40	-15,3767	-17,8817	-17,8817	-18,6186	-22,3447
41	-17,3948	-20,559	-20,559	-25,0325	-28,5109
42	-13,0487	-19,9055	-22,6379	-22,9135	-25,8799
43	-19,7857	-23,4034	-23,4034	-23,4034	-25,3849
44	-17,1143	-21,3204	-21,3204	-21,3204	-24,0196
45	-15,205	-15,5991	-15,5991	-15,9897	-17,6146

As Tabelas (4.5) e (4.6) apresentam os valores de $p - Valor_{MC}$ para os testes de Cascavel, onde cada teste em que $p - Valor_{MC} \leqslant 5\%$ foi marcado em vermelho. O teste com $\delta_S = 10$ e $\delta_T = 0$, por exemplo, apresenta clusterização espaço-temporal, enquanto o teste com $\delta_S = 10$ e $\delta_T = 3$ não.

Tabela 4.5: Primeira parte dos valores do pValor das simulações de MC nos testes realizados com dados de Cascavel, onde valores menores que 5% estão destacados em coloração avermelhada.

TCrit.	SCrit. 10	SCrit. 30	SCrit. 50	SCrit. 75	SCrit. 100
0	0,011	0,023	0,176	0,023	0,048
1	0,266	0,356	1	0,069	0,107
2	0,176	0,342	0,279	0,359	0,416
3	1	1	1	1	1
4	0,279	0,366	0,19	0,373	0,47
5	1	1	1	1	0,452
6	0,19	0,247	1	0,315	0,387
7	1	1	1	1	1
8	1	1	0,187	1	1
9	1	1	1	1	0,491
10	0,187	0,295	0,235	0,329	0,459
11	1	0,026	1	0,046	0,018
12	0,235	0,441	1	0,452	0,153
13	1	0,484	1	0,503	0,516
14	1	1	1	1	1
15	1	1	1	1	1
16	1	1	1	1	1
17	1	1	1	1	1
18	1	1	1	1	1
19	1	1	1	1	1
20	1	1	0,186	1	1
21	1	1	1	1	1
22	0,186	0,253	1	0,265	0,385

Tabela 4.6: Segunda parte do pValor das simulações de MC.

TCrit.	SCrit. 10	SCrit. 30	SCrit. 50	SCrit. 75	SCrit. 100
23	1	1	1	1	0,294
24	1	1	1	1	1
25	1	1	1	1	1
26	1	1	1	1	1
27	1	0,276	1	0,31	0,371
28	1	1	0,29	1	1
29	1	1	1	0,321	0,482
30	0,29	0,293	1	0,317	0,548
31	1	1	1	1	1
32	1	1	1	1	1
33	1	1	1	1	1
34	1	1	1	1	1
35	1	1	1	1	0,287
36	1	1	1	1	1
37	1	1	1	1	1
38	1	1	1	1	1
39	1	1	1	1	1
40	1	1	1	1	1
41	1	1	1	1	1
42	1	1	1	1	1
43	1	1	1	1	1
44	1	1	0,023	1	1
45	1	1	0,356	1	1

As Tabelas (4.7)-(4.9) apresentam os dados na noção de ER, conforme formulações de Aldstadt (2007). Os números e as cores nestas são referentes às categorias listadas na Tabela (4.10).

Tabela 4.7: Parte 1 da tabela de ER de Cascavel.

100	2	2	2	1	2	2	2	1	1	2	2	2	2	2	1	1	1	1	1
75	2	2	2	1	2	1	2	1	1	1	2	2	2	2	1	1	1	1	1
50	2	2	2	1	2	1	2	1	1	1	2	2	2	2	1	1	1	1	1
30	2	2	2	1	2	1	2	1	1	1	2	3	2	2	1	1	1	1	1
10	2	2	2	1	2	1	2	1	1	0	3	1	3	0	1	1	0	1	0
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18

Tabela 4.8: Parte 2 da tabela de ER de Cascavel.

100	1	1	1	2	2	1	1	1	2	1	2	2	1	1	1	1	2
75	1	1	1	2	1	1	1	1	2	1	2	2	0	1	1	1	1
50	1	1	1	2	1	1	1	0	3	1	2	3	0	1	0	1	1
30	1	1	1	2	1	1	1	0	3	0	1	4	0	0	0	1	1
10	0	0	1	3	0	0	0	0	1	0	1	5	0	0	0		1
	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35

Tabela 4.9: Parte 3 da tabela de ER de Cascavel.

100	1	1	1	1	1	1	1	1	1	1
75	1	1	1	1	1	1	1	1	1	1
50	1	1	1	1	1	1	1	0	1	1
30	0	0	0	1	1	1	1	0	1	1
10	0	0	0	1	1	1	1	0	0	0
	36	37	38	39	40	41	42	43	44	45

Tabela 4.10: Legenda das tabelas ER, onde os intervalos são do tipo "[Inferior, Superior)".

Categoria	Inferior	Superior	Cor
0	0	0,9	
1	0,9	1	
2	1	1,15	
3	1,15	1,3	
4	1,3	1,6	
5	1,6	+INF	

4.1.2 Caso Rio de Janeiro

A Figura 4.4 mostra os pontos de ocorrência no Rio de Janeiro distribuídos espacialmente, e a Figura 4.5 mostra a distribuição das estatísticas em cada testes iKnox executado. Diferentemente do caso anterior, o máximo global de cada distribuição espacial ao longo do tempo situa-se quando $\delta_T = 1$, ou seja, o maior número de casos clusterizados em qualquer espaço testado ocorreram com um dia de diferença.

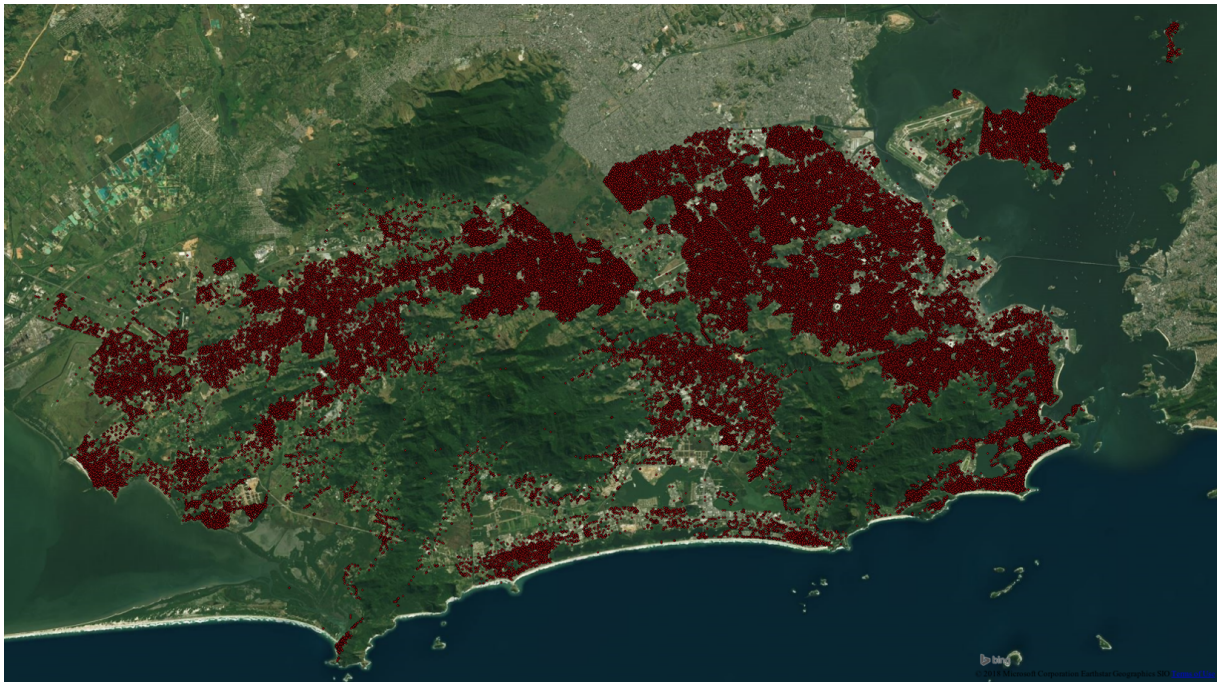

Figura 4.4: Distribuição espacial dos casos do Rio de Janeiro.

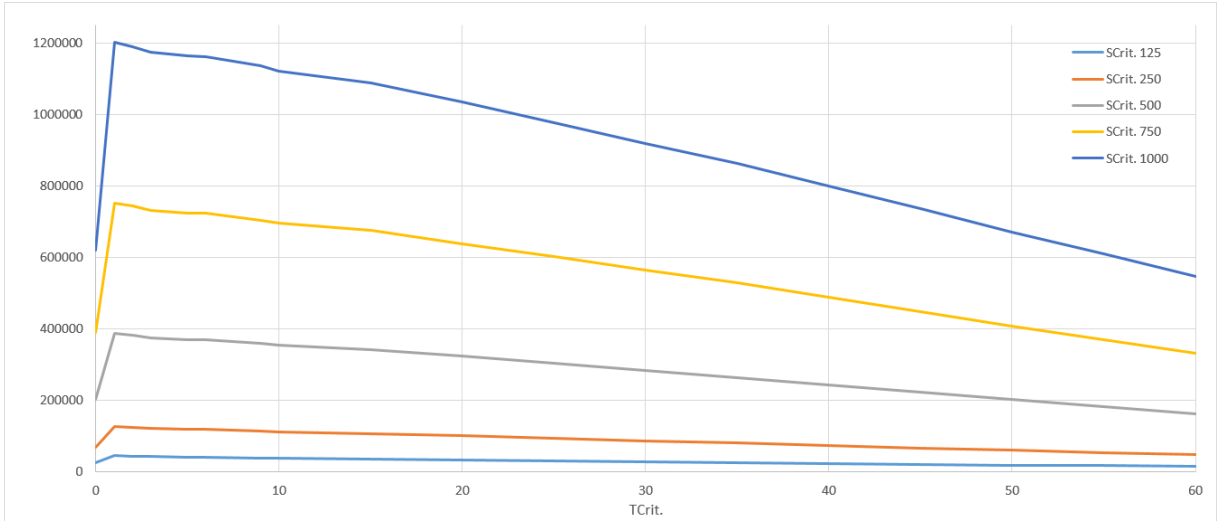

Figura 4.5: Distribuição da estatística de iKnox nos testes do Rio de Janeiro onde o eixo Y é referente aos valores da estatística, o eixo X do tempo crítico e cada linha é um espaço crítico.

A Tabela (4.11) apresenta os dados utilizados para a construção do gráfico da Figura 4.5.

Tabela 4.11: Valores da estatística de iKnox nos testes realizados com dados do Rio de Janeiro.

TCrit.	SCrit. 125	SCrit. 250	SCrit. 500	SCrit. 750	SCrit. 1000
0	26742	68426	202444	389976	620867
1	46176	127794	388084	751691	1201678
2	44497	124664	382881	743368	1190334
3	42865	121363	375218	732414	1175237
5	41603	119301	370468	724064	1164331
6	40817	118646	369625	723035	1162195
9	39160	115040	359753	705218	1135816
10	38450	112421	354437	697120	1121868
15	36458	108143	342598	675180	1088391
20	33908	101377	323873	638824	1034586
25	31706	94856	303752	602378	977782
30	28612	87214	283656	565035	918851
35	26196	80815	264564	529473	863067
40	24016	74376	244000	489796	801253
45	21581	67692	223166	449463	736747
50	19595	60794	202139	408283	671065
55	17479	55119	183625	370833	610029
60	15654	49282	164074	331922	547012

Similar ao caso anterior, a Figura 4.6 apresenta a distribuição dos valores normalizados das estatísticas dos 90 testes e a Tabela (4.12) apresenta os dados explícitos desse gráfico.

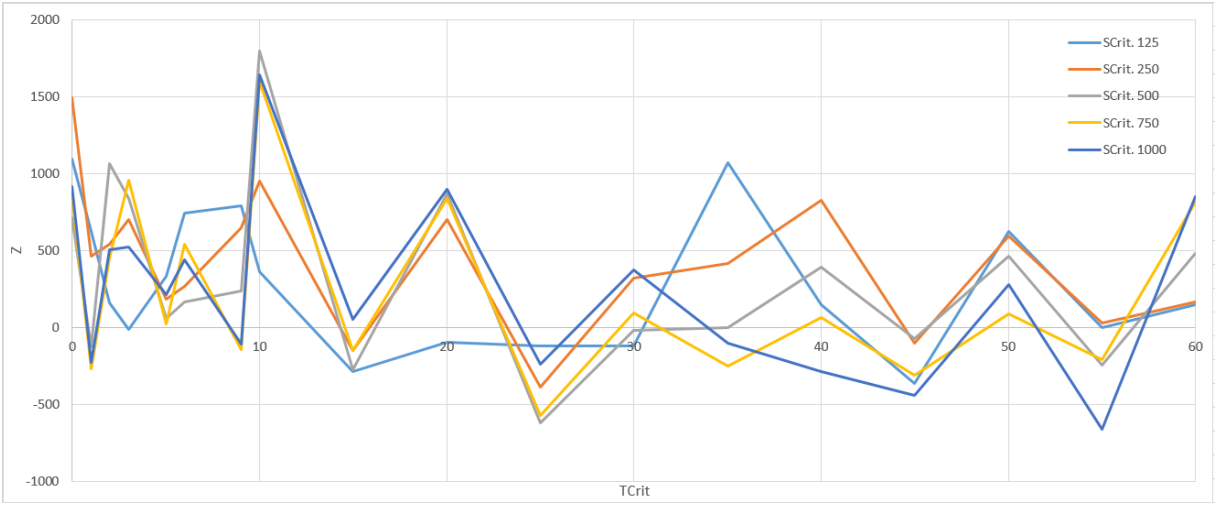

Figura 4.6: Distribuição dos valores de Z para a estatística de iKnox nos testes do Rio de Janeiro onde o eixo Y é referente aos valores de Z, o eixo X do tempo crítico e cada linha é um espaço crítico

Tabela 4.12: Valores de Z nos testes realizados com dados do Rio de Janeiro.

TCrit.	SCrit. 125	SCrit. 250	SCrit. 500	SCrit. 750	SCrit. 1000
0	1096,065	1496,206	718,5856	864,939	919,9666
1	628,3836	464,9758	-125,461	-266,941	-223,4052
2	162,6801	541,279	1068,594	451,8787	509,3306
3	-12,5332	701,2526	843,6844	958,4769	524,18957
5	332,1439	186,5504	59,86551	27,28174	215,68126
6	745,8728	266,811	167,5488	545,7858	439,0126
9	794,0364	650,8487	237,8213	-142,558	-104,6273
10	366,5674	951,5237	1801,875	1605,562	1645,8075
15	-287,056	-146,03	-275,316	-146,82	53,038937
20	-91,1472	706,2886	869,4378	841,6823	902,16036
25	-120,658	-387,086	-615,7	-572,871	-238,5731
30	-118,708	323,56	-18,1005	96,56346	377,61233
35	1075,267	420,8159	-1,2762	-246,326	-97,69146
40	151,5262	831,6319	392,1967	63,91146	-286,5731
45	-361,666	-100,763	-69,6338	-308,429	-439,1948
50	625,3099	598,319	465,6125	89,43848	280,42778
55	-0,24987	31,95105	-241,677	-204,232	-659,3738
60	150,4668	167,7696	485,7378	818,2939	855,31679

Por fim, a Tabela (4.13) apresenta os valores de $p - Valor_{MC}$ com os resultados menores ou iguais à 5% assinalados em vermelho. Nestes, observa-se que houve clusterização espaço-temporal com $\delta_S = 500$ e $\delta_T = 10$, por exemplo.

Tabela 4.13: Valores do pValor das simulações de MC nos testes realizados com dados do Rio de Janeiro, onde valores menores que 5% estão destacados em coloração avermelhada.

TCrit.	SCrit. 125	SCrit. 250	SCrit. 500	SCrit. 750	SCrit. 1000
0	0,055	0,009	0,107	0,06	0,048
1	0,187	0,214	0,64	0,753	0,706
2	0,473	0,196	0,033	0,212	0,151
3	0,651	0,129	0,066	0,043	0,165
5	0,354	0,396	0,492	0,497	0,343
6	0,141	0,358	0,413	0,161	0,207
9	0,11	0,162	0,359	0,644	0,641
10	0,337	0,053	0,001	0,003	0,001
15	0,837	0,699	0,756	0,653	0,503
20	0,72	0,111	0,06	0,056	0,047
25	0,725	0,862	0,932	0,91	0,712
30	0,722	0,297	0,559	0,445	0,253
35	0,042	0,263	0,552	0,733	0,611
40	0,466	0,075	0,24	0,458	0,757
45	0,882	0,643	0,588	0,767	0,85
50	0,145	0,137	0,192	0,443	0,291
55	0,572	0,512	0,707	0,68	0,925
60	0,453	0,41	0,167	0,053	0,04

4.2 Aspectos Computacionais

A proposta apresentada possui relativas melhorias em relação às duas utilizadas como base de comparação, Tango (2010) e Höhle et al. (2017). Estas podem ser separadas em melhorias sobre a complexidade de espaço e de tempo, e podem ser resumidas na Tabela (4.14).

Tabela 4.14: Resumo das otimizações da solução proposta.

Memória	Tempo
Representação dos dados (bit-a-bit)	Algoritmos mais rápidos baseados na representação
Particionamento em blocos	Variância: O(n^3) para O(n^2)
Paralelismo interno às simulações de MC	Trechos paralelizados

Quanto às do espaço, a representação dos dados é referente às adjacências, que foram reduzidas de 4 bytes, tamanho de um valor lógico em R, para 1 bit cada, trazendo uma redução de 32 vezes. O particionamento em blocos é para que não seja necessário manter todos os dados, casos e\ou adjacências, em memória, podendo o código ser executado com entradas grandes e em sistemas com pouca memória principal, sem os impactos de desempenho que ocorreriam

com a recuperação de paginas realizadas pelo sistema operacional, precisando-se apenas ajustar o tamanho dos blocos utilizados. A última melhoria listada é o paralelismo interno às simulações de MC, que é diferente do utilizado por Höhle et al. (2017), em outras palavras, a solução proposta executa blocos, partes, das matrizes em paralelo, não sobre cópias das mesmas. Para exemplificar, considera-se um sistema com quatro *threads* em um teste com o paralelismo interno e outro com externo, no primeiro teriam-se quatro blocos de adjacências temporais e no segundo teriam-se quatro matrizes completas. Observa-se que essa diferença do espaço utilizado por conta da representação não é exata, pois os blocos podem ser parciais, em que certas posições são preenchidas com valores zero, para não ter influência sobre o resultado. Nota-se também que apesar da solução proposta apresentar melhorias no aspecto de quantia de dados representados se comparar com a solução de Tango (2010), esta não foi listada por ser similar à segunda implementação utilizada na comparação.

Para as do tempo, o primeiro ponto é sobre a possibilidade do uso de algoritmos voltados à representação bit-a-bit tratando múltiplos casos por vez, trazendo melhorias no desempenho aplicados às etapas de contagem nos trechos de estatística observada, variância e simulações de MC. O segundo é sobre a melhoria na complexidade para cálculo dos fatores n_2, reduzida da ordem cúbica para quadrática. O último ponto é sobre onde o paralelismo está presente. Enquanto no primeiro código para comparação listado não há paralelismo algum, o segundo apresenta nas simulações MC contra a solução proposta, que aplica paralelismo, interno aos blocos como supracitado, em todos os trechos supracitados além do cálculo das distâncias.

Com esse pretexto, listam-se os resultados obtidos com os testes das implementações nas Tabelas (4.15) a (4.17). Estes foram feitos sobre a implementação de Tango (2010), a implementação de Höhle et al. (2017) e a solução proposta implementada em C. As duas últimas, apresentam testes com apenas uma *thread* e com quatro para comparar com o código não paralelizado e o paralelismo entre as mesmas.

Observa-se que (t), presente na Tabela (4.15), é referente ao teste em que a execução foi manualmente abortada devido ao tempo relativamente grande para a obtenção dos resultados. Nas Tabelas (4.16) e (4.17), (s) é o ponto em que o ambiente utilizado para os testes, "R Studio" para o código em R, não executou pois não conseguiu realizar a alocação de memória, sendo necessário mais que os 16 GB disponíveis. Para as mesmas tabelas $(s)*$ é o teste que iniciou

mas foi manualmente abortado pois estourou o limite de memória do sistema. A coluna MC apresenta 3 variações, em que "Não" é relativa aos testes sem simulação de MC, utilizando a expressão (2.14) para o $p - Valor$ aproximado e "99" e "999" são para testes com simulação de MC com as respectivas quantias de amostragem, utilizando a expressão (2.16) para o valor simulado.

Quanto às possíveis escolhas de parâmetros ou algoritmos para a solução proposta que foi feita na linguagem "C", cujos testes estão sob as colunas "*Single thread*" e "*Multithread*", destaca-se que os testes sem simulações MC utilizaram a estratégia "por linhas" para o cálculo dos termos n_2, enquanto os com MC não realizaram essa etapa, já que a variância pode ser calculada com base nas simulações. Algo similar vale para os testes dos códigos de Tango (2010), em que os sem MC o código com n_2 cúbico foi utilizado e com MC o código sem nenhum n_2 foi utilizado. Para os testes que $n \leqslant 200$, o tamanho do bloco foi de 50 e para testes em que $n > 200$, o tamanho foi de 250. As escolhas dos tamanhos foi completamente empírica já que 250 apresentou desempenho satisfatório em vários testes, exceto nos menores, em que 50 apresentou desempenho temporal semelhante, mas ocupando $2Mb$ a menos.

Tabela 4.15: Comparações do pico máximo de memória, em KBytes, e desempenho, em segundos, da implementação de Tango (2010).

n	MC	Memória	Tempo
50	Não	2.029,4	24,14
	99	34.871,8	369,94
	999	465.122,2	3.713,73
100	Não	58.468	388,38
	99	339.793,6	2.370,95
	999	3.786.388,1	24.364,48
200	Não	456.911,5	2.990,44
	99	1.669.433,1	9.718,78
	999	(t)	(t)

Tabela 4.16: Comparações do pico máximo de memória, em KBytes, das diferentes implementações, onde "Surv." é para a implementação de Höhle et al. (2017).

		Single thread		Multithread	
n	MC	Surv.	Proposta	Surv.	Proposta
50	Não	3.610	3.300	3.126	3.428
	99	3.861	3.332	3.994	3.488
	999	3.917	3.344	4.096	3.468
100	Não	3.980	3.280	3.174	3.432
	99	3.992	3.340	3.277	3.476
	999	4.051	3.344	4.301	3.460
200	Não	5.063	3.296	4.198	3.432
	99	5.152	3.340	4.506	3.472
	999	5.236	3.376	5.222	3.476
500	Não	5.536	5.292	5.427	5.412
	99	5.648	5.267	7.168	5.456
	999	5.782	5.288	8.090	5.416
1.000	Não	8.909	5.534	8.909	5.672
	99	50.630	5.546	19.149	5.712
	999	69.031	5.599	19.968	5.924
2.500	Não	98.406	6.484	99.533	6.648
	99	118.374	6.488	1.178.060	6.600
	999	232.476	6.500	1.074.376	6.612
5.000	Não	687.488	7.893	394.445	7.976
	99	905.395	7.963	1.970.095	7.904
	999	1.582.732	7.954	2.746.514	8.020
10.000	Não	1.748.840	10.474	1.567.539	10.636
	99	2.514.846	10.549	8.115.720	10.544
	999	2.530.693	10.510	8.428.738	10.544
20.000	Não	6.428.964	15.569	6.264.320	15.440
	99	9.452.102	15.569	(s)*	15.456
	999	9.490.348	15.532	(s)	15.448
50.000	Não	(s)	31.298	(s)	30.824
	99	(s)	31.322	(s)	30.824
	999	(s)	31.281	(s)	30.840
100.000	Não	(s)	57.590	(s)	56.496
	99	(s)	57.594	(s)	56.500
	999	(s)	57.614	(s)	56.524
225.704	Não	(s)	123.572	(s)	120.956
	99	(s)	123.609	(s)	120.960
	999	(s)	123.585	(s)	120.996

Tabela 4.17: Comparações de desempenho, em segundos, das diferentes implementações, onde "Surv." é para a implementação de Höhle et al. (2017).

		Single thread		Multithread	
n	MC	Surv.	Proposta	Surv.	Proposta
50	Não	0,11	0	0,12	0
	99	0,13	0	5,82	0
	999	0,21	0,04	6,08	0,05
100	Não	0,11	0,01	0,14	0
	99	0,13	0,02	5,88	0,01
	999	0,28	0,12	6,08	0,08
200	Não	0,12	0	0,14	0
	99	0,16	0,03	5,98	0,02
	999	0,61	0,13	6,10	0,23
500	Não	0,13	0,02	0,14	0
	99	0,42	0,14	6,16	0,08
	999	3,45	1,48	7,59	0,75
1.000	Não	0,14	0,03	0,26	0,01
	99	1,67	0,36	7,25	0,16
	999	14,68	3,15	15,11	1,49
2.500	Não	0,26	0,20	0,69	0,07
	99	16,30	1,16	16,30	0,50
	999	157,02	9,66	90,92	4,37
5.000	Não	1,08	0,80	1,08	0,26
	99	76,74	3,24	45,41	1,23
	999	751,36	25,41	391,00	10,64
10.000	Não	2,57	3,14	2,56	0,99
	99	332,78	10,11	184,62	3,51
	999	3.333,92	81,32	1.660,31	30,91
20.000	Não	8,52	14,01	8,50	3,84
	99	1.408,96	39,15	(s)*	11,98
	999	14.265,12	281,49	(s)	99,15
50.000	Não	(s)	91,24	(s)	28,44
	99	(s)	239,81	(s)	78,70
	999	(s)	1.589,10	(s)	551,23
100.000	Não	(s)	371,24	(s)	117,55
	99	(s)	944,87	(s)	317,10
	999	(s)	5.823,95	(s)	2.191,15
225.704	Não	(s)	1.859,10	(s)	591,73
	99	(s)	4.443,78	(s)	1.666,01
	999	(s)	30.553,27	(s)	11.569,23

Para a mais fácil comparação, foram feitos gráficos das tabelas de memória e desempenho comparando as diferentes implementações, todos com escala logarítmica com base 10. A Figura

4.7 compara o uso de memória de todas as implementações testadas. Nota-se que a proposta apresenta apenas as entradas sem MC pois o uso de memória é muito próximo nos três tipos de teste. A Figura 4.8, por sua vez, apresenta o pico de memória para os testes *Multithread*. Nota-se que como não foram testados todos os tamanhos de entrada n para todas as implementações devido ao tempo grande de execução ou falta de memória, como supracitado, algumas linhas são interrompidas, como todas as variações de testes Tango (2010), por exemplo.

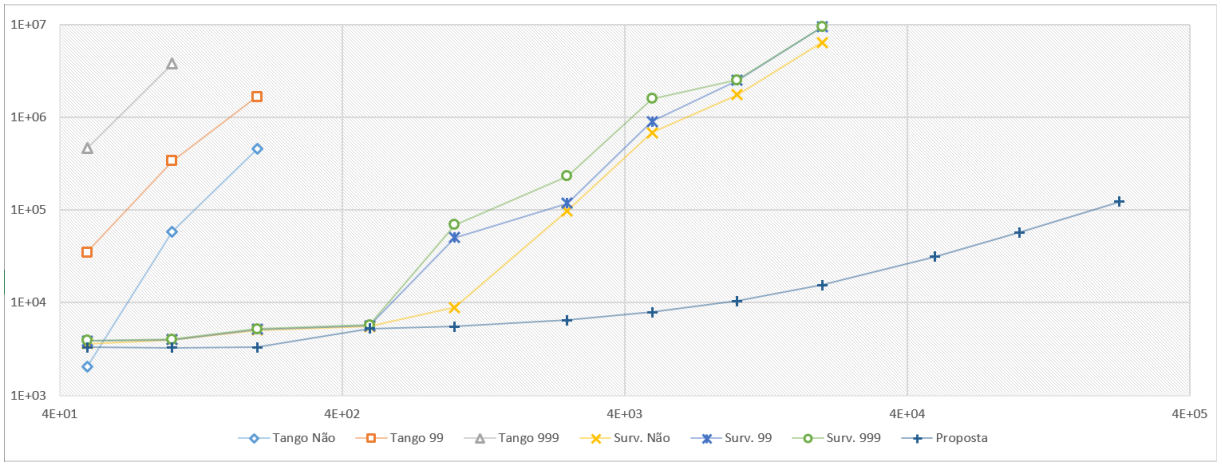

Figura 4.7: Gráfico de pico de memória das implementações *single thread* testadas, com Kbytes no eixo Y e n no eixo X.

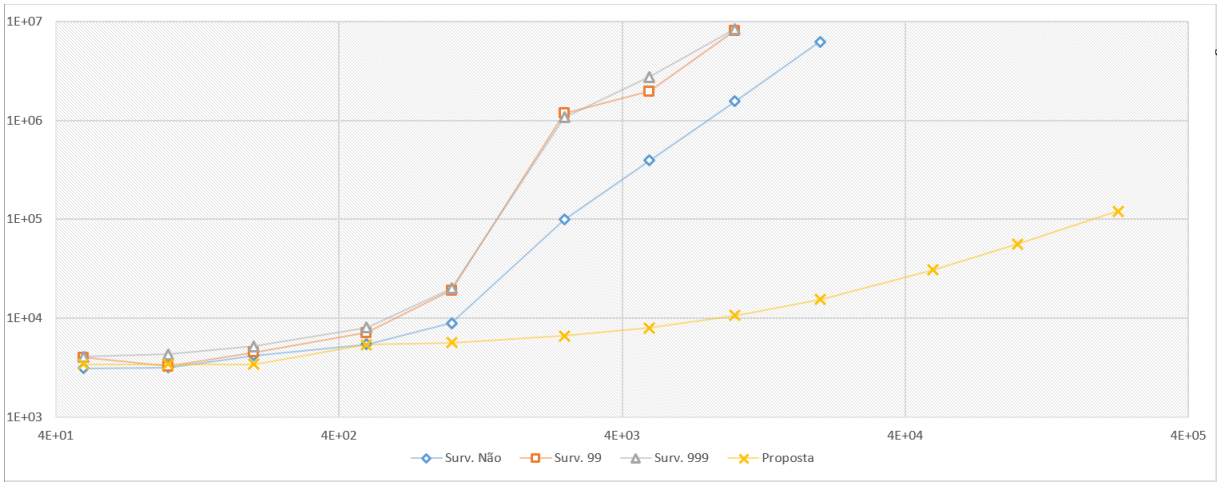

Figura 4.8: Gráfico de pico de memória das implementações *multithread* testadas, com Kbytes no eixo Y e n no eixo X.

Em relação ao tempo de execução, a Figura 4.9 apresenta os dados para os testes *sSingle thread* e a Figura 4.10 para os *multithread*.

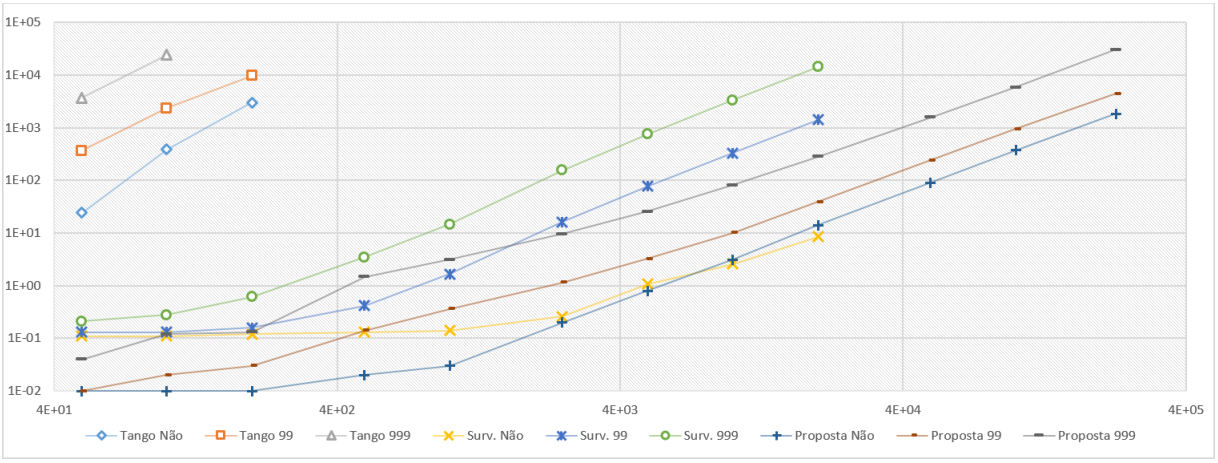

Figura 4.9: Gráfico do tempo de execução das implementações *single thread* testadas, com segundos no eixo Y e n no eixo X.

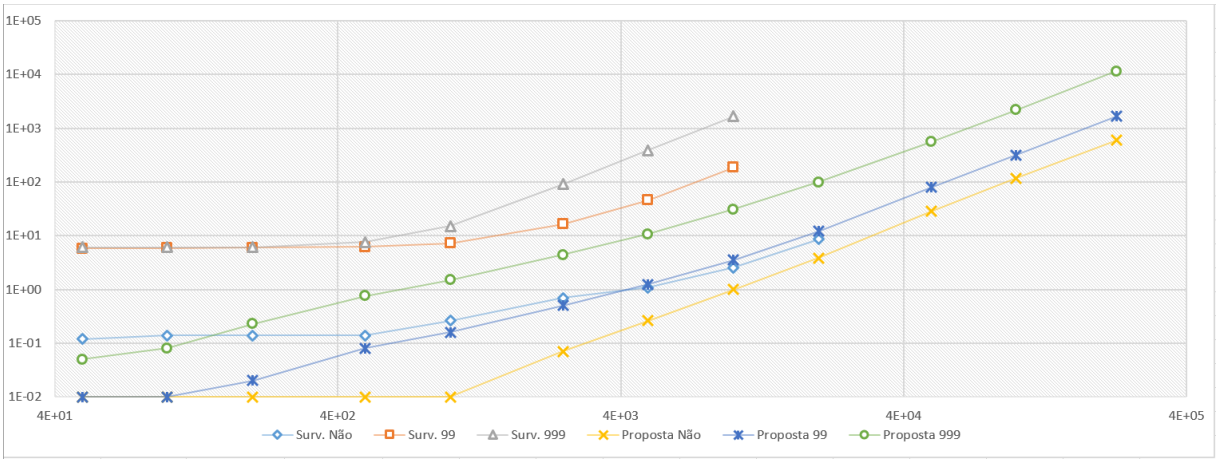

Figura 4.10: Gráfico do tempo de execução das implementações *multithread* testadas, com segundos no eixo Y e n no eixo X.

Além da comparação entre todas as implementações, há a comparação entre a escalabilidade com os testes *multithread*. A Figura 4.11 apresenta comparação de pico de memória da implementação de (HöHLE et al., 2017) e a Figura 4.12 a comparação do tempo de execução.

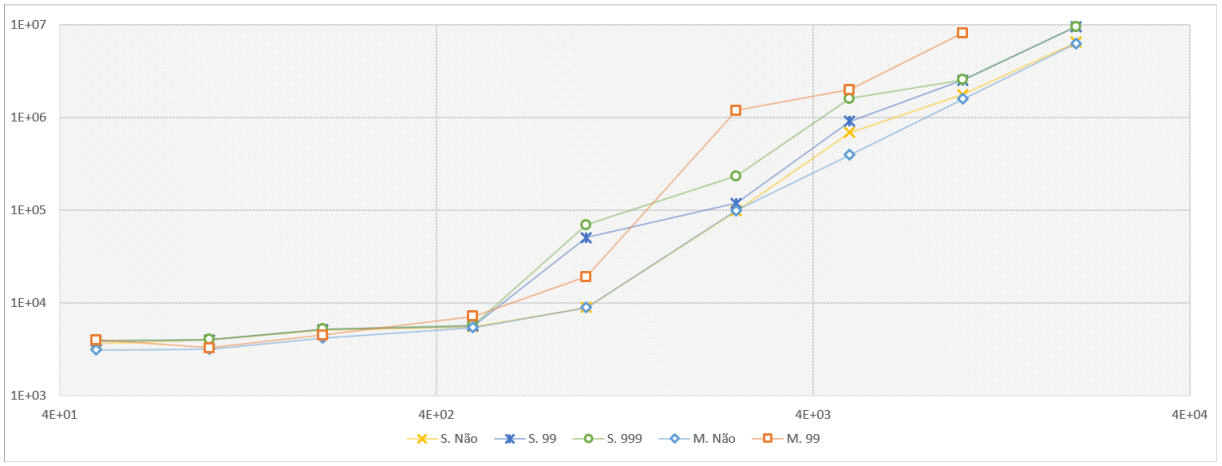

Figura 4.11: Gráfico de pico de memória da implementação de Höhle et al. (2017), *single vs. multithread*, com Kbytes no eixo Y e n no eixo X.

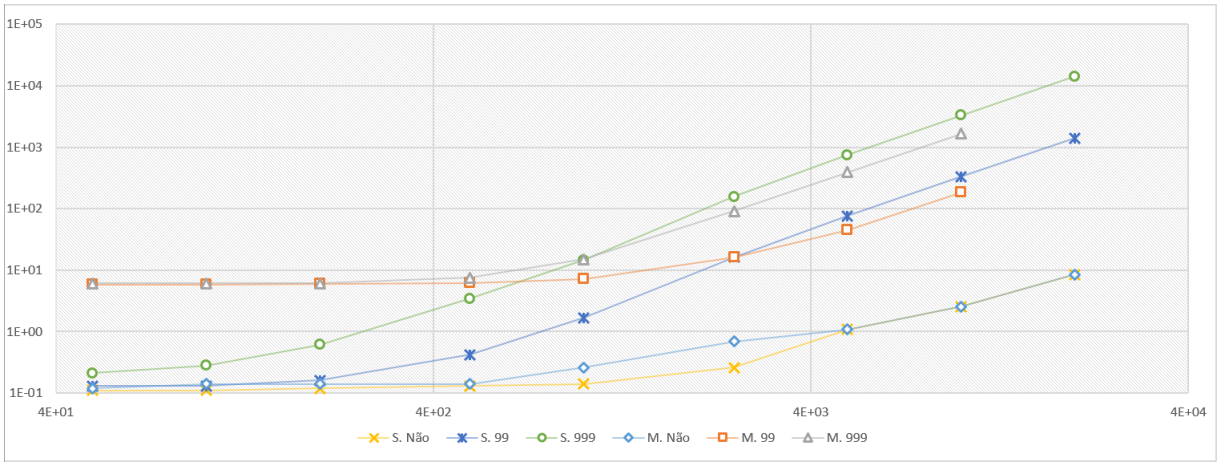

Figura 4.12: Gráfico de tempo de execução da implementação de Höhle et al. (2017), *single vs. multithread*, com segundos no eixo Y e n no eixo X.

Similarmente, a Figura 4.13 apresenta a comparação do tempo de execução da solução proposta. Nota-se que, devido diferença do pico de memória entre os testes da solução proposta ser ínfima, este gráfico não é apresentado.

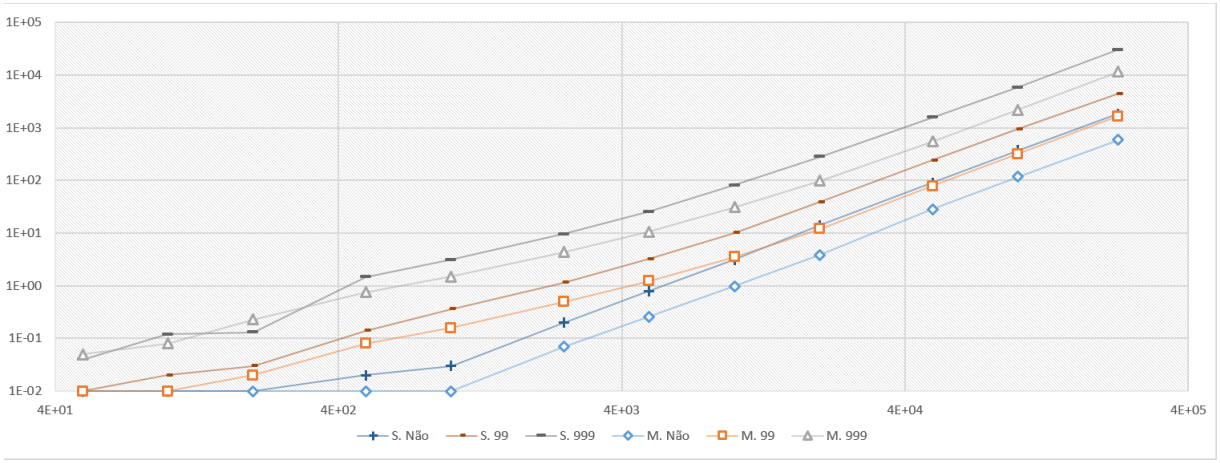

Figura 4.13: Gráfico de pico de memória da implementação proposta, *single vs. multithread*, com segundos no eixo Y e n no eixo X.

Pelas tabelas e figuras anteriores, nota-se que a solução por Höhle et al. (2017) tende a utilizar mais memória que a proposta quando n tende a crescer, isto é, possui mais complexidade espacial. Ademais, nota-se que há um crescimento substancial no uso de memória no primeiro código quando há simulações de MC, o que não ocorre no segundo. Algo similar pode ser notado para os casos onde há o uso de paralelismo.

Em relação ao tempo e como no caso do espaço, a melhoria da solução proposta para as comparadas torna-se mais evidente conforme n é maior. Observando ambas as colunas de tempo de "Surv.", é notável que o paralelismo é aplicado somente na etapa de MC, pois apresenta melhor desempenho apenas quando este é presente. No paralelismo da solução proposta, por sua vez, é possível notar que o paralelismo é aplicado também quando não há simulações MC.

Em termos de desempenho temporal relativo, o código "Surv." apresentou, no melhor caso, $2,01$ de *speedup* relativo na comparação de paralelismo com $n = 10.000$ e 999 simulações MC. A solução proposta teve o melhor caso com $n = 20.000$ e sem simulações de MC, com um *speedup* relativo de $3,648$. No desempenho espacial, houveram execuções em que primeiro código ocupa 80% mais memória quando executado com 4 *threads*, como com $n = 2.500$ e 999 simulações. O pior caso para a solução proposta, por sua vez, foi com $n = 1.000$ e o mesmo número de simulações, em que a execução paralela ocupou cerca de 6% mais memória que sua versão serial.

Capítulo 5

Conclusões

Neste trabalho foi apresentado o método de Knox de maneira geral e sua variação incremental. Ademais foram apresentadas melhorias para as implementações existentes em relação à complexidade espacial e ao tempo de execução. Por fim, apresentou-se estudos de caso para dados de Dengue em Cascavel e no Rio de Janeiro, bem como comparações de desempenho espaço-temporal entre diversos n, ora com 1 *thread* ora com 4 *threads*. Inclusive, foi implementada uma solução da abordagem iKnox, não disponibilizada nos códigos analisados.

Para casos em que a quantidade de casos de entrada são suficientemente grandes, em geral $n \geqslant 20.000$ com 4 *threads* e 999 simulações de Monte Carlo, a solução proposta viabilizou não somente a execução da clusterização que anteriormente não era possível, mas comparativamente às situações em que era viável sua execução, ganhos significativos foram obtidos no tempo de execução e no espaço máximo utilizado. Se considerar $n = 10.000$, 4 *threads* e 999 simulações MC, por exemplo, a implementação de (HöHLE et al., 2017) teve um pico de cerca de $8.428.738$ Kbytes contra 10.554 KBytes da proposta, totalizando aproximadamente $800x$ de ganho. No mesmo teste, a implementação testada levou $1.660,31$ segundos para finalizar a execução, contra $30,91$ da proposta, uma redução de $53,71$ vezes. Similarmente para $n = 5.000$, a proposta apresentou uma melhoria $2.746.514/8.020 \approx 342,46x$ no pico de memória e $391/10,64 \approx 36,75x$ no tempo necessário para a finalização.

Para trabalhos futuros, sugere-se:

- Uma implementação onde as adjacências são armazenadas no formato *bitstring*, reduzindo a memória utilizada e possivelmente reduzindo o tempo de execução;
- Testes com os tamanhos e parâmetros utilizados em máquinas com maiores quantias de

threads, para testar a escalabilidade da solução;

- Testes em FPGA e GPUs, para testar a escalabilidade da solução em plataformas que visam extrair o máximo de paralelismo, com um exemplo de solução FPGA apresentado no Apêndice C;

- Testes para verificar possíveis diferenças nas simulações de MC resultantes de como a amostragem é aplicada na a solução proposta em comparação à de Meyer (2017).

Apêndice A

Simulações de Monte Carlo utilizadas nas seções de exemplo

Neste apêndice, listam-se todos os valores utilizados para todas as simulações de Monte Carlo para as seções em que foram apresentados exemplos, como 2.1.2, 2.2.1 e 3.6. Primeiramente listam-se as simulações no formato da seção 2.1.2 e depois as de MC para todos os testes da seção 2.2.1, com códigos para evitar a repetição de Tabelas. Nota-se que para os exemplos de amostragem, o método utilizado para as mesmas é o "*shift*-circular", onde cada amostra é um *shift* da anterior.

As Tabelas A.1 e A.2 são para todas as simulações de MC cujos resultados estão presentes na Tabela (2.7), observe que além dos dados temporais amostrados, que vão do $MC : 1$ até $MC : 9$, há os observados em $MC : 0$.

Tabela A.1: Dados observados e simulações 0 à 4, com S para o vetor com adjacências espaciais, T temporais e ST ambos.

		MC: 0		MC: 1		MC: 2		MC: 3		MC: 4	
Dupla	S	T	ST	T	ST	T	ST	T	ST	T	ST
1	0	1	0	1	0	0	0	1	0	0	0
2	0	1	0	1	0	1	0	0	0	1	0
3	0	0	0	1	0	1	0	1	0	0	0
4	1	0	0	0	0	1	1	1	1	1	1
5	0	1	0	0	0	0	0	1	0	1	0
6	0	0	0	1	0	0	0	0	0	1	0
7	0	0	0	0	0	1	0	0	0	0	0
8	1	1	1	0	0	0	0	1	1	0	0
9	0	0	0	1	0	0	0	0	0	1	0
10	0	1	0	0	0	1	0	0	0	0	0
Total	2	5	1	5	0	5	1	5	2	5	1

Tabela A.2: Simulações 5 à 9 de MC, com S para o vetor com adjacências espaciais, T temporais e ST ambos.

		MC: 5		MC: 6		MC: 7		MC: 8		MC: 9	
Dupla	S	T	ST	T	ST	T	ST	T	ST	T	ST
1	0	0	0	1	0	0	0	0	0	1	0
2	0	0	0	0	0	1	0	0	0	0	0
3	0	1	0	0	0	0	0	1	0	0	0
4	1	0	0	1	1	0	0	0	0	1	1
5	0	1	0	0	0	1	0	0	0	0	0
6	0	1	0	1	0	0	0	1	0	0	0
7	0	1	0	1	0	1	0	0	0	1	0
8	1	0	0	1	1	1	1	1	1	0	0
9	0	0	0	0	0	1	0	1	0	1	0
10	0	1	0	0	0	0	0	1	0	1	0
Total	2	5	0	5	2	1	1	5	1	5	1

A Tabela A.3 contém os valores observados para os limiares $\delta_S = \{0,5; 1; 1,5\}$. Esta última apresenta apenas 3 Tabelas pois não há variação nas simulações de MC para informações do espaço.

Tabela A.3: Tabela iKnox com as adjacências para os três limiares espaciais utilizados em teste e, na última linha, os valores para n_s.

δ_S		
0,5	1	1,5
0	0	0
0	0	0
0	0	0
0	1	1
0	0	0
0	0	0
0	0	0
1	1	1
0	0	1
0	0	1
1	2	4

As Tabelas A.4, A.5 e A.6, por sua vez, apresentam os valores observados e simulados, totalizando dez vetores, para $\delta_T = \{1; 2; 3\}$, respectivamente, e os valores para n_t e n_{st}, três vezes repetidos para cada comparação com os limiares espaciais.

Tabela A.4: Tabela iKnox com $\delta_T = 1$, em que cada coluna é uma amostra de MC, com 0 sendo a observada, e apresenta resultados de n_t e n_{st}, combinando com cada δ_S.

MC	0	1	2	3	4	5	6	7	8	9
	0	1	0	1	0	0	1	0	0	1
	1	0	1	0	1	0	0	1	0	0
	0	1	0	1	0	1	0	0	1	0
	0	0	1	0	1	0	1	0	0	1
	1	0	0	1	0	1	0	1	0	0
	0	1	0	0	1	0	1	0	1	0
	0	0	1	0	0	1	0	1	0	1
	1	0	0	1	0	0	1	0	1	0
	0	1	0	0	1	0	0	1	0	1
	1	0	1	0	0	1	0	0	1	0
n_t	4	4	4	4	4	4	4	4	4	4
0,5	1	0	0	1	0	0	1	0	1	0
1	1	0	1	1	1	0	2	0	1	1
1,5	2	1	2	1	2	1	2	1	2	2

Tabela A.5: Tabela iKnox com $\delta_T = 2$, em que cada coluna é uma amostra de MC, com 0 sendo a observada, e apresenta resultados de n_t e n_{st}, combinando com cada δ_S.

MC	0	1	2	3	4	5	6	7	8	9
	0	0	1	0	0	1	0	0	1	0
	0	0	0	1	0	0	1	0	0	1
	1	0	0	0	1	0	0	1	0	0
	0	1	0	0	0	1	0	0	1	0
	0	0	1	0	0	0	1	0	0	1
	1	0	0	1	0	0	0	1	0	0
	0	1	0	0	1	0	0	0	1	0
	0	0	1	0	0	1	0	0	0	1
	1	0	0	1	0	0	1	0	0	0
	0	1	0	0	1	0	0	1	0	0
n_t	3	3	3	3	3	3	3	3	3	3
0,5	0	0	1	0	0	1	0	0	0	1
1	0	1	1	0	0	2	0	0	1	1
1,5	1	2	1	1	1	2	1	1	1	1

Tabela A.6: Tabela iKnox com $\delta_T = 3$, em que cada coluna é uma amostra de MC, com 0 sendo a observada, e apresenta resultados de n_t e n_{st}, combinando com cada δ_S.

MC	0	1	2	3	4	5	6	7	8	9
	0	0	0	0	1	0	0	1	0	0
	0	0	0	0	0	1	0	0	1	0
	0	0	0	0	0	0	1	0	0	1
	1	0	0	0	0	0	0	1	0	0
	0	1	0	0	0	0	0	0	1	0
	0	0	1	0	0	0	0	0	0	1
	1	0	0	1	0	0	0	0	0	0
	0	1	0	0	1	0	0	0	0	0
	0	0	1	0	0	1	0	0	0	0
	0	0	0	1	0	0	1	0	0	0
n_t	2	2	2	2	2	2	2	2	2	2
0,5	0	1	0	0	1	0	0	0	0	0
1	1	1	0	0	1	0	0	1	0	0
1,5	1	1	1	1	1	1	1	1	0	0

Finalmente, a Tabela A.7 é um resumo de apenas as Tabelas observadas e codificações e a Tabela (A.8) apresenta como as Tabelas anteriores são relacionadas para os nove testes da seção de exemplo do *iKnox* (e.g. n_{st} calculado utilizando as colunas "a"e "x" situaria-se na célula esquerda-inferior da tabela ER).

Tabela A.7: Resumo das adjacências espaço-temporais observadas do exemplo de iKnox, onde as letras na base são usadas para referenciar cada vetor de adjacência na tabela ER posterior.

Espacial			Temporal		
0,5	1	1,5	1	2	3
0	0	0	0	0	0
0	0	0	1	0	0
0	0	0	0	1	0
0	1	1	0	0	1
0	0	0	1	0	0
0	0	0	0	1	0
0	0	0	0	0	1
1	1	1	1	0	0
0	0	1	0	1	0
0	0	1	1	0	0
a	b	c	x	y	z

Tabela A.8: Relação das Tabelas de adjacências anteriores quando colocadas em uma tabela ER, com S para os valores espaciais e T temporais.

3	a & z	b & z	c & z
2	a & y	b & y	c & y
1	a & x	b & x	c & x
T \ S	0,5	1,0	1,5

Apêndice B

Modelo dos fluxogramas utilizados

O padrão dos fluxogramas é o da figura B.1.

Figura B.1: Modelo dos fluxogramas.

Apêndice C

Arquitetura para FPGAs

Nesse apêndice é apresentada uma proposta de arquitetura para o teste de Knox em uma plataforma híbrida CPU+FPGAs. Nesta proposta, visou-se minimizar a quantia de dados transferidos entre a memória principal e o FPGA, consequentemente, escolheu-se calcular as distâncias e amostragens no processador, restando o cálculo da quantia de adjacências para o FPGA.

Primeiramente, as distâncias entre as duplas são calculadas, preferencialmente em blocos e em paralelo. Para facilitar, o tamanho dos blocos sempre foi escolhido como um múltiplo de 8, pois endereçamento é a nível de byte, armazenando 8 duplas por byte. Após calculadas as distâncias para as duplas do bloco em memória, armazenam-se as adjacências bit-a-bit. Considerando um bloco espacial e outro temporal, faz-se um $bitwiseAND$ e manda-se o bloco resultante para o FPGA, que retorna a quantia de adjacências. Para as simulações de MC, faz-se a amostragem de cada bloco em processador enquanto o FPGA calcula a contagem do bloco da simulação anterior. Quando todos os blocos forem processados, realiza-se uma etapa de pós-processamento para obter dados necessários, como, por exemplo, a média das simulações de MC.

Com essa abordagem, um teste com $n = 200.000$ teria aproximadamente 20 bilhões de bits para processar em FPGA, precisando retornar 35 bits para a estatística observada e cada simulação de MC caso todo o processamento fosse executado de uma só vez. Como não é possível, os resultados parciais devem ser armazenados, implicando em 1.000 contadores adicionais para um teste com 999 simulações.

O componente do FPGA foi elaborado com uma estratégia "dividir para conquistar" cujo elemento mínimo é uma LUT (*look-up-table*) para 4 bits, ou seja, a entrada, que é múltipla de 4, é dividida na metade até que seu tamanho seja 4, que é então calculada utilizando uma LUT

somando todos os resultados parciais com somadores até obter a contagem do bloco de bits de entrada.

C.1 Exemplo

Apresenta-se aqui um exemplo de execução para o cálculo das principais métricas do teste de Knox em um componente de FPGA de exemplo que realiza a contagem sobre 8 bits, utilizando os mesmos dados da Seção 3.6. Nota-se que as métricas não calculadas no exemplo são os termos n_2, que precisariam de uma estruturação de matriz ou com contadores, conforme discutido na Seção 3.2.

Como o componente de exemplo é de 8 bits, montam-se blocos de adjacência de 1 byte. Primeiramente, calculam-se as distâncias e proximidades de 8 casos e armazenam-se elas em dois bytes, um para espacial e outro temporal. Realiza-se um $bitwiseAND$ entre os dois blocos e o resultante é enviado para o componente, que retorna a contagem parcial, como na Tabela (C.1). Depois, realiza-se uma amostragem, como na Tabela (C.2), sobre o bloco, que é então enviado para o processamento pelo componente. Realizam-se as amostragens restantes, armazenando os resultados parciais em contadores, passando para o próximo bloco ao fim dessa etapa.

Tabela C.1: Primeiro bloco a ser enviado ao componente, cujo o resultado é armazenado no contador 0, referente à estatística observada.

Espacial	0	0	0	1	0	0	0	1	
Temporal	1	1	0	0	1	0	0	1	Contador 0
p/ o componente	0	0	0	0	0	0	0	1	1

Tabela C.2: Primeira amostragem do primeiro bloco, cujo resultado retornado pelo componente é armazenado no contador 1, referente à simulação 1.

Espacial	0	0	0	1	0	0	0	1	
Temporal	1	1	1	0	0	1	0	0	Contador 1
p/ o componente	0	0	0	0	0	0	0	0	0

Observa-se o último bloco pode ser parcial, como nesse exemplo, em que há apenas 2 elementos, como na Tabela (C.3), necessitando de uma amostragem somente sobre as posições válidas. Apesar das posições inválidas não entrarem na amostragem, são marcadas com 0 para que não alterem o resultado final quando o bloco for processado pelo componente, que não

precisa fazer a distinção de posições válidas ou inválidas. Na mesma tabela, o contador a ser atualizado é o 0 pois esse bloco ainda não foi amostrado nenhuma vez e o valor de 1 é por causa do resultado parcial obtido pela passagem do primeiro bloco pelo componente.

Tabela C.3: Último bloco, onde as posições inválidas são delimitadas por -" com o intuito ilustrativo.

Espacial	0	0	-	-	-	-	-	-	
Temporal	0	1	-	-	-	-	-	-	Contador 0
p/ o componente	0	0	0	0	0	0	0	0	1

Referências Bibliográficas

ALDSTADT, J. *Spatial and spatiotemporal analysis of dengue virus transmission and Aedes aegypti abundance*. Tese (Doutorado) — University of California, Santa Barbara, 2007.

BAKER, R. D. Testing for space-time clusters of unknown size. *Journal of Applied Statistics*, Taylor & Francis, v. 23, n. 5, p. 543–554, 1996.

BOZA, J. A. R. *Agregaciones espacio-temporales. Test de Knox.* [S.l.]: Federación de Enseñanza de CC.OO. de Andalucía, 2009. Consultado na INTERNET:https://www.feandalucia.ccoo.es/docuipdf.aspx?d=4822&s=, 2018. (Temas para la Educación, v. 2).

DAVID, F.; BARTON, D. Two space-time interaction tests for epidemicity. *British journal of preventive & social medicine*, BMJ Group, v. 20, n. 1, p. 44, 1966.

GEAR, J. C. *A test for detecting space-time clustering and a comparison with some existing methods*. Tese (Doutorado) — Dissertation to the faculty of the University of North Carolina., 2006.

HöHLE, M. et al. *surveillance: Temporal and Spatio-Temporal Modeling and Monitoring of Epidemic Phenomena*. 2017. Consultado na INTERNET: https://cran.r-project.org/web/packages/surveillance/index.html, 2017.

JACQUEZ, G. M. Spatial cluster analysis. *S. Fotheringham and J. Wilson (Eds.)*, Blackwell Publishing, p. 395–416, 2008.

KALAROT, R.; MORRIS, J. Comparison of fpga and gpu implementations of real-time stereo vision. In: IEEE. *Computer Vision and Pattern Recognition Workshops (CVPRW), 2010 IEEE Computer Society Conference on*. [S.l.], 2010. p. 9–15.

KNOX, E.; BARTLETT, M. The detection of space-time interactions. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, JSTOR, v. 13, n. 1, p. 25–30, 1964.

KNOX, G. Epidemiology of childhood leukaemia in northumberland and durham. *British journal of preventive & social medicine*, BMJ Group, v. 18, n. 1, p. 17, 1964.

KNUTH, D. E. *The art of computer programming, 3rd edn. seminumerical algorithms, vol. 2*. [S.l.]: Addison-Wesley, Reading, 1997. 145-146 p.

LIAO, C. et al. Early experiences with the openmp accelerator model. In: SPRINGER. *International Workshop on OpenMP*. [S.l.], 2013. p. 84–98.

MANTEL, N. The detection of disease clustering and a generalized regression approach. *Cancer research*, AACR, v. 27, n. 2 Part 1, p. 209–220, 1967.

MCLAFFERTY, S. Disease cluster detection methods: recent developments and public health implications. *Annals of GIS*, GIS, p. 127–133, 2015.

MEYER, S. *knox.R. Knox test for space-time interaction. Part of the surveillance package, http://surveillance.r-forge.r-project.org*. 2017. Consultado na INTERNET: https://github.com/jimhester/surveillance/blob/master/R/knox.R, 2015.

NORTH, B. V.; CURTIS, D.; SHAM, P. C. *A Note on the Calculation of Empirical P Values from Monte Carlo Procedures*. 2003. Consultado na INTERNET: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC529334/pdf/AJHGv72p496s2.pdf.

OLIVERIA M. A.; RIBEIRO, H. C.-S. Geospatial analysis applied to epidemiological studies of dengue: a systematic review. *Revista Brasileira de Epidemiologia*, p. 907–917, 2013.

PIKE, M.; SMITH, P. Disease clustering: a generalization of knox's approach to the detection of space-time interactions. *Biometrics*, JSTOR, p. 541–556, 1968.

PRESS, W. H. et al. *Numerical recipes in C*. [S.l.]: Cambridge university press Cambridge, 1996.

RATHER, I. A. e. a. Prevention and control strategies to counter dengue virus infection. *Frontiers in Cellular and Infection Microbiology*, 2017.

SILVA, F. R. *Desenvolvimento de uma medida de Associação entre Espaço e Tempo*. Tese (Doutorado) — Departamento de Estatística. Instituto de Ciências Exatas. Universidade Federal de Minas Gerais. Mestado em Estatística., 2011.

TANGO, T. *Statistical methods for disease clustering*. [S.l.]: Springer Science & Business Media, 2010.

TERASIC. *DE2i-150 FPGA Development Kit*. 2017. Consultado na INTERNET: http://www.terasic.com.tw/cgi-bin/page/archive.pl?Language= English&CategoryNo=11&No=529>, 2016.

VICENTI-GONZALEZ, M. E. e. a. Spatial analysis of dengue seroprevalence and modeling of transmission risk factors in a dengue hyperendemic city of venezuela. *PLOS Neglected Tropical Diseases 11*, PLOS, 2017.

WHO., W. H. O. *Dengue and Severe Dengue*. 2016. Consultado na INTERNET: http://www.who.int/wer/2016/wer9130.pdf, 2018.

WHO., W. H. O. *Dengue and Severe Dengue*. 2018. Consultado na INTERNET: http://www.who.int/news-room/fact-sheets/detail/dengue-and-severe-dengue, 2018.

WHO., W. H. O. *Dengue control*. 2018. Consultado na INTERNET: http://www.who.int/denguecontrol/control_strategies/en/, 2018.