Avaliação da Performance Paralela do Método dos Conjuntos Análogos Usando KD-Tree e Implementado em Pthreads*

Pedro Henrique de Paula Assis†, Claudio Schepke

[1]Laboratório de Estudos Avançados em Computação (LEA)
Universidade Federal do Pampa (UNIPAMPA) - Alegrete - RS - Brazil

{pedroassis}.aluno@unipampa.edu.br
{claudioschepke}@unipampa.edu.br

Resumo. *O Método dos Conjuntos Análogos é utilizado para reconstruir séries temporais incompletas usando séries correlacionadas. No entanto, seu custo computacional pode ser substancial, devido à inclusão de diversas variáveis ao longo de longos períodos de treinamento. Este artigo demonstra que a paralelização permite processar grandes bases climáticas em estações de trabalho comuns. A implementação alcançou um speedup de 7,06x com 12 threads.*

1. Introdução

O Método dos Conjuntos Análogos (*Analog Ensemble* - AnEn) é uma técnica estatística para previsão probabilística e reconstrução de séries temporais meteorológicas [Clementino et al. 2026b]. O método baseia-se na busca por cenários históricos que mais se assemelham às condições atuais [Clementino 2025]. Sua importância reside na alta precisão para estimar incertezas com baixo custo computacional comparado a modelos físicos numéricos, sendo essencial para previsões de energias renováveis e agricultura.

Embora eficiente, o AnEn enfrenta um gargalo na busca em grandes volumes de dados [Clementino et al. 2026a]. O algoritmo processa extensas bases históricas, como dados meteorológicos coletados por longos períodos de tempo. Para cada ponto de previsão, exigem-se milhões de cálculos envolvendo padrões de distância. Embora existam outras abordagens na literatura que buscam paralelizar o AnEn, a exploração eficiente da memória compartilhada em estações de trabalho convencionais ainda carece de avaliações práticas aprofundadas.

Este artigo avalia uma proposta de otimização paralela para o método *Analog Ensemble* (AnEn), utilizando a estrutura de dados KD-Tree [Bentley 1975] e a biblioteca Pthreads [IEEE 1995] em linguagem C para acelerar a busca de análogos em séries históricas meteorológicas. A principal novidade desta proposta frente aos trabalhos anteriores do grupo, reside na alocação da árvore inteiramente em memória compartilhada para acesso exclusivo de leitura (*read-only*), eliminando travamentos (*locks*) e permitindo escalar o desempenho em processadores multicore de forma mais eficiente.

2. Implementação Paralela do Método dos Conjuntos Análogos

A Figura 1 ilustra o funcionamento do método Analog Ensemble (AnEn), cujo processo pode ser dividido em três etapas principais. Na primeira etapa (1), com base nas condições

*Trabalho parcialmente financiado pelo Edital CNPq: Projeto 135963/2023-0.
†Bolsista PRO-IC/Unipampa 2025.

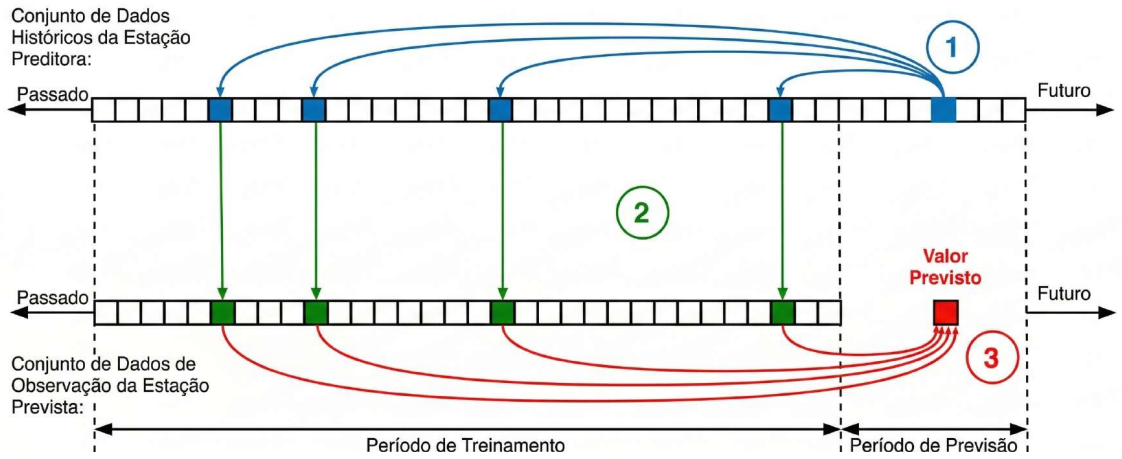

Figura 1. O método dos Conjuntos Análogos (adaptado de [Breve et al. 2024]).

atuais ou previstas de uma Estação Preditora, realiza-se uma busca em todo o seu conjunto de dados históricos (passado) para encontrar as janelas de tempo que mais se assemelham à condição atual, formando o conjunto de análogos. Na etapa seguinte (2), os instantes de tempo exatos em que esses análogos ocorreram são mapeados para a base de dados de observação da Estação Prevista. Por fim, na etapa (3), os valores reais observados nesses períodos históricos são extraídos e combinados (formando o *ensemble*) para calcular o Valor Previsto final.

O grande desafio computacional desse método reside na Etapa 1. Encontrar as janelas mais semelhantes exige o cálculo de distâncias matemáticas entre o cenário atual e milhões de registros históricos, contendo múltiplas variáveis climáticas simultâneas (como temperatura, pressão e vento). Para evitar uma busca linear exaustiva, que seria extremamente lenta, o sistema utiliza a estrutura de dados KD-Tree (Árvore K-Dimensional). A KD-Tree particiona esse espaço multidimensional de dados de forma hierárquica. Assim, em vez de comparar a condição atual com cada registro do passado um por um, o algoritmo realiza uma busca otimizada de vizinhos mais próximos (*Nearest Neighbor Search*). Vale ressaltar que a paralelização foca estritamente nesta primeira etapa, uma vez que as etapas seguintes (mapeamento e extração) possuem um custo computacional irrisório e alta dependência sequencial que não justificam o *overhead* de criação de *threads*.

O código sequencial original, foi desenvolvido em trabalhos prévios e modificado com a inclusão da biblioteca Pthreads [Clementino et al. 2026a]. A principal alteração algorítmica consistiu em particionar o laço principal de busca de análogos (decomposição de domínio), alocando um subconjunto de consultas independente para cada *thread* processar. A estrutura da KD-Tree foi alocada em memória compartilhada apenas para leitura, o que evitou condições de corrida e travamentos (*locks*), permitindo um ganho linear de velocidade sem qualquer perda na precisão matemática original do modelo. Essa abordagem de baixo nível minimiza o *overhead* de sincronização e maximiza o uso dos núcleos do processador, garantindo controle total sobre a gestão de memória e latência.

3. Avaliação Experimental - Resultados

Para validar a performance, a proposta de avaliação integra as informações de dados climáticos brutos (em formato NetCDF) com algoritmos de busca espacial eficiente. Através da construção de uma árvore k-dimensional com particionamento cíclico espacial, o sistema organiza o histórico de dados (2011-2019) da velocidade do vento. Para fins de reprodutibilidade, as consultas à KD-Tree foram configuradas para retornar os 25 vizinhos mais próximos ($k = 25$), utilizando vetores com dimensionalidade 41 (janela de tempo definida por $2K + 1$, com $K = 20$), permitindo que múltiplas *threads* processem consultas simultâneas de forma coordenada em toda a extensão da série histórica.

A arquitetura de processamento paralelo em memória compartilhada está detalhada na Tabela 1. O ambiente experimental utiliza um processador AMD Ryzen 5 9600X (6 núcleos e 12 threads), apoiado por 16GB de memória RAM DDR5 a 6000MHz, o que minimiza a latência de acesso aos dados durante a busca de análogos. Além disso, o uso de um SSD SATA III (550MB/s de leitura) para o armazenamento do sistema e da base de dados garante uma vazão estável de I/O, permitindo que o processador opere próximo de sua capacidade máxima, sem gargalos significativos de leitura.

Tabela 1. Especificações do Hardware Utilizado

Componente	**Especificação**
Processador (CPU)	AMD Ryzen 5 9600X (6C/12T)
Memória RAM	16GB DDR5 6000MHz CL30
Armazenamento	SSD SATA III (550MB/s)
Placa-Mãe	ASUS TUF Gaming B650
Sistema Operacional	Ubuntu 24.04.3 LTS, versão do kernel 6.17.0-14-generic

A Tabela 2 apresenta os resultados da avaliação de performance realizada na arquitetura previamente descrita. A avaliação foca no monitoramento do tempo de execução, *speedup* e eficiência em cenários de execução de 1 a 12 threads, visando identificar o limite de saturação do hardware e garantir que a distribuição de carga não afete a precisão científica dos resultados.

Tabela 2. Resultados de Performance e Escalabilidade (Média de 5 execuções para o Período 2011-2019)

Threads	**Tempo (s) ± DP**	**Speedup**	**Eficiência (%)**	**Qualidade (Erro)**
1	901,21 ± 3,09	1,00x	100,0%	4,5096
2	446,92 ± 1,12	2,02x	100,8%	4,5096
4	227,72 ± 1,11	3,96x	98,9%	4,5096
8	159,86 ± 0,35	5,64x	70,5%	4,5096
12	127,64 ± 0,84	7,06x	58,8%	4,5096

Para garantir a validade estatística, cada cenário de paralelismo foi executado 5 vezes de forma isolada, gerando o tempo médio e o desvio padrão (DP) apresentados na Tabela 2. Os resultados demonstram um ganho de performance expressivo para o processamento de *big data* ambiental. A utilização de 12 *threads* no processador Ryzen 9600X resultou no menor tempo médio de execução (127,64s), alcançando um *speedup*

máximo de 7,06×. Observa-se uma escalabilidade quase ideal até 4 *threads*, mantendo uma eficiência próxima a 99%. A partir de 8 *threads*, a eficiência começa a decair, refletindo o *overhead* natural de sincronização e o limite imposto pelas seções sequenciais do código (Lei de Amdahl). Ainda assim, o tempo total de processamento foi reduzido drasticamente de aproximadamente 15 minutos para pouco mais de 2 minutos, sem qualquer alteração na precisão dos resultados preditivos. A métrica de qualidade constante de 4,5096 reportada representa a Distância Euclidiana Média entre o vetor alvo de previsão e o respectivo análogo mais próximo retornado pela árvore.

4. Conclusão

Este trabalho demonstrou a eficácia da paralelização do método estatístico Analog Ensemble (AnEn) utilizando a biblioteca Pthreads e a estrutura espacial KD-Tree. A abordagem implementada solucionou o gargalo computacional das buscas por análogos em vastas séries históricas, indicando que é possível alcançar alta escalabilidade e eficiência em processadores multicore modernos. O fator mais relevante do estudo foi a capacidade de reduzir drasticamente o tempo de resposta do modelo preditivo para a amostra avaliada, preservando a mesma precisão matemática da implementação sequencial original.

Como perspectivas para trabalhos futuros, visando o processamento de bases climáticas em escala global e de altíssima resolução, sugere-se a migração desta arquitetura para aceleradores de hardware (GPUs), utilizando interfaces como CUDA ou OpenACC, aproveitando as técnicas de particionamento e gestão de memória já validadas neste projeto.

Referências

Bentley, J. L. (1975). Multidimensional Binary Search Trees Used for Associative Searching. *Comm. of the ACM*, 18(9):509–517.

Breve, M., Camargos, A., Rufino, J., and Balsa, C. (2024). Computational Performance Analysis of PCA Enhanced AnEn Method. In *Proceedings of the Workshop on Applications of Computational Mathematics to Simulation and Data Analysis (ACMaSDA 2024)*, page 1833. Santiago de Chile, Chile.

Clementino, A., Schepke, C., Balsa, C., and Rufino, J. (2026a). A Multithreaded Implementation of the Analog Ensemble Method Assisted by a k-d Tree. In *2026 34rd Euromicro International Conference on Parallel, Distributed, and Network-Based Processing (PDP)*.

Clementino, A., Schepke, C., Balsa, C., and Rufino, J. (2026b). Using k-d Trees to Leverage the Performance of the Analog Ensemble Method. In Guarda, T., Portela, F., and Augusto, M. F., editors, *Advanced Research in Technologies, Information, Innovation and Sustainability*, pages 32–47, Cham. Springer Nature Switzerland.

Clementino, A. R. (2025). Parallelization of the Analog Ensemble Algorithm in Pthreads. Master's thesis, School of Technology and Management of Bragança (ESTiG) - Informatics Engineering.

IEEE (1995). IEEE Std 1003.1c-1995: Information Technology - Portable Operating System Interface (POSIX) - Part 1: System Application Program Interface (API) Amendment 2: Threads Extension [C Language]. Standard, Institute of Electrical and Electronics Engineers.