Classificação de Tempo de Execução de Jobs de Simulação em HPC Usando LLMs e Engenharia de Prompt

Luciana de C. Franci[1], Lucas A. F. da Costa[1], Leonardo V. Neri[1], Jeffson Carneiro Silva Simões[1], Felipe A. Portella[2]

[1] Centro de Estudos e Sistemas Avançados do Recife (CESAR), Recife – Brasil

[2] Petróleo Brasileiro S.A. (PETROBRAS), Rio de Janeiro – Brasil

{lcf2,lacf,lvn,jcss}@cesar.org.br, lucasalexandre27@gmail.com, felipeportella@petrobras.com.br

Resumo. *Este trabalho propõe o uso de Large Language Models (LLMs) e Engenharia de Prompt para classificar o tempo de execução de simulações de reservatórios em ambientes HPC, visando otinizar a alocação de recursos. Foram comparadas três estratégias (Ingênuo, Especialista e Few-Shot) utilizando 10.000 scripts. A abordagem Few-Shot, que incorporou estatísticas históricas, obteve o melhor desempenho com 76,83% de acurácia, superando métodos tradicionais ao interpretar a semântica dos códigos sem execução prévia. Embora promissores, os resultados em classes desbalanceadas indicam que futuras melhorias exigem estratégias de ajuste fino ou aumento de dados para aumentar a robustez da classificação em cenários complexos.*

1. Introdução

A previsão e a otimização da simulação de reservatórios dependem de modelos numéricos detalhados, cuja resolução apresenta desafios técnicos associados à complexidade matemática das equações governantes, que frequentemente acoplam fluxo multifásico, termodinâmica composicional e efeitos geomecânicos [Meng *et al.*, 2025]. A necessidade de maior fidelidade na representação das heterogeneidades geológicas resulta em malhas altamente refinadas, gerando sistemas de equações não lineares cujo custo computacional limita a viabilidade do uso de arquiteturas de *hardware* tradicionais. Nesse contexto, a aplicação da Computação de Alto Desempenho (*High-Performance Computing* - HPC) consolida-se como a abordagem padrão na engenharia de reservatórios. A infraestrutura de HPC permite o particionamento do domínio e a paralelização da resolução dos sistemas algébricos, reduzindo o tempo de processamento por meio da distribuição de carga em processadores *multicore* e em arranjos de aceleradores gráficos (GPUs) [Middya *et al.*, 2021]. Contudo, a execução eficiente dessas tarefas exige estratégias algorítmicas de ajuste fino (*tuning*) focadas em otimizar as cargas de trabalho (*workloads*) computacionais para a arquitetura disponível [Portella *et al.*, 2022]. Esse processamento otimizado viabiliza tanto a avaliação de modelos geológicos detalhados quanto a execução de fluxos baseados em múltiplas realizações, suportando abordagens avançadas de quantificação de incertezas e ajuste de histórico em tempo hábil para a tomada de decisão [Tuczyński and Stopa, 2023].

Prever com precisão o tempo de execução dos scripts de submissão (*job*) em HPC é crucial para escalonamento, *backfilling* e melhor uso de recursos. Preditores tradicionais utilizam campos dos *jobs* e parâmetros do gerenciador de recursos (por

exemplo, Slurm) combinados com modelos como florestas aleatórias (*random forests*), *gradient boosting*, redes neurais profundas (*DNNs*) ou aprendizado amplo (*broad learning*) [Wang et al. 2021]. A adoção de uma estratégia de classificação, que categoriza a duração dos *jobs* em intervalos de tempo predefinidos, tem demonstrado oferecer maior robustez para minimizar erros percentuais nas estimativas e facilitar a tomada de decisão do escalonador [Nunes et al. 2025a]. Trabalhos recentes mostram que com o uso de LLMs é possível ir além dos métodos clássicos, explorando scripts, código e logs, não ficando limitado aos modelos de aprendizado de máquina clássicos. A abordagem com LLMs baseia-se na sua capacidade nativa de analisar e extrair a semântica de características textuais contidas nos scripts (como parâmetros e binários selecionados) e associá-las aos metadados do *job*. Mediante o desenvolvimento de templates de prompt específicos e o uso de estratégias de Aprendizado em Contexto, como a apresentação de amostras do histórico de *jobs* com as diferenças estruturais destacadas (*diff-based*), o modelo é guiado a compreender mapeamentos complexos e padrões sutis, oferecendo uma predição categórica mais assertiva para apoiar o planejamento eficiente das plataformas HPC [Liu et al. 2025]. Com o objetivo de investigar abordagens para superar as limitações de execução dos *jobs*, que são intrínsecas aos modelos de simulação de reservatórios, este trabalho propõe a aplicação de LLMs integrados a técnicas de Engenharia de Prompt para a classificação do tempo de execução de *jobs* de simulação em HPC.

2. Metodologia

Para avaliar a capacidade de LLMs na classificação do tempo de execução de *jobs* de simulação em HPC, utilizou-se um conjunto de dados composto por 10.000 scripts de submissão de *jobs*, acompanhados de metadados sobre o simulador utilizado e o número de blocos ativos no modelo de reservatório. O tempo de execução real de cada *job* foi categorizado em quatro classes ordinais: Curto (300–1800 s), Intermediário (1801–7200 s), Longo (7201–86400 s) e Super Longo (acima de 86400 s). O modelo GPT-4.1, acessado via Azure OpenAI, foi integrado por meio do framework LangChain [Chase 2022] com saída estruturada utilizando os parâmetros de temperatura igual a 0.1 e *top_p* igual a 0.1. O prompt de inferência é composto pelo contexto de uma estratégia, o texto do script e uma descrição para gerar duas predições: a classificação mais provável e uma alternativa em caso de evidência ambígua. O processamento das inferências foi paralelizado para viabilizar a execução em escala do conjunto de dados.

Foram comparadas três estratégias de engenharia de prompt: (i) prompt Ingênuo (*Naive*), fornecendo apenas o script e os metadados sem contexto adicional, baseado na técnica *Zero-Shot Prompting* [Kojima et al. 2022]; (ii) prompt Especialista de Domínio, incorporando conhecimento de domínio sobre comportamento típico de cada simulador e limiares de blocos ativos associados a cada classe de tempo; e (iii) prompt Few-Shot, que acrescenta estatísticas históricas, calculadas a partir dos dados históricos por simulador e classe de tempo de execução, incluindo tempo médio, média e desvio-padrão do número de blocos ativos, bem como a frequência relativa de cada classe.

O desempenho de cada abordagem foi avaliado por meio de matriz de confusão, relatório de classificação (precisão e F1-score nas médias macro e ponderada), acurácia

combinada (considerando acerto na predição primária *ou* alternativa) e métricas Erro Absoluto Médio (MAE) e Erro Percentual Absoluto médio (MAPE). A fim de comparar experimentalmente com modelos de aprendizado de máquina, o conjunto de dados foi dividido em treinamento (80%) e teste (20%) para treinar um modelo XGBoost, adaptando o número de classes e *features* do método descrito em Nunes *et al.* 2025b.

3. Resultados e Discussão

Os experimentos em 10.000 scripts de submissão de *jobs* HPC (Tabela 1) demonstram que a incorporação progressiva de contexto eleva substancialmente o poder preditivo do modelo. Enquanto os prompts Ingênuo e Especialista de Domínio apresentaram limitações de desempenho, o prompt Few-Shot obteve os ganhos mais expressivos da análise.

Tabela 1. Comparação de desempenho das três estratégias de *prompting*.

Métrica	Ingênuo	Especialista	*Few-Shot*	*XGBoost*
Métricas globais				
Acurácia	43,07%	69,65%	76,83%	**88,85%**
Erros completos	56,93%	30,35%	**23,17%**	-
MAE	1,09	0,44	**0,23**	5047,69
MAPE	92,88%	43,02%	**20,87%**	72,90%
Precisão (média ponderada)	0,45	0,76	0,85	**0,89**
Recall (média ponderada)	0,43	0,7	0,77	**0,88**
F1-score (média ponderada)	0,39	0,62	0,75	**0,88**
F1-score (média macro)	0,30	0,46	0,58	**0,79**
F1-score por classe				
Curto	0	0,26	**0,6**	-
Intermediário	0,60	**0,84**	0,79	-
Longo	0,61	0,75	**0,92**	-
Super longo	0	0	0	-
Consumo do modelo GPT-4.1 por *job*				
Tokens (mediana)	707	1126	1337	-
Custo (USD) (média)	$0,0016	$0,0027	$0,0030	-
Latência (segundos) (média)	0,93	0,89	0,959	-

Ao adicionar distribuições históricas de tempo de execução e blocos ativos, a abordagem Few-Shot atingiu o melhor desempenho global (Tabela 1). A Figura 1 ilustra a progressão das matrizes de confusão das três abordagens, evidenciando o

fortalecimento da diagonal principal do prompt Ingênuo ao Few-Shot, o que indica melhoria na acurácia e na capacidade discriminativa. O modelo tradicional XGBoost apresentou acurácia de 88,85% e F1-score de 0,88, mas índices elevados de MAE e MAPE indicando desvios acentuados nas predições incorretas; em contrapartida, a abordagem Few-Shot obteve 76,83% de acurácia e MAE e MAPE baixos, sugerindo estabilidade analítica capaz de manter as margens de erro em níveis controlados.

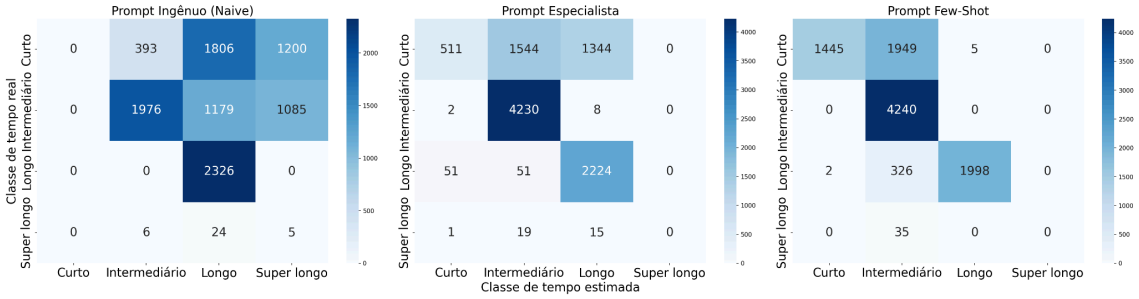

Figura 1. Matrizes de confusão comparativas de três estratégias de *prompting*.

O aumento de 33,76 pontos percentuais na acurácia combinada no prompt Few-Shot demonstra a viabilidade de LLMs contextualizados para a predição de tempo de execução em HPC. Todavia, o desbalanceamento amostral resultou em F1-score nulo para a classe Super longo em todas as abordagens. Adicionalmente, o F1-score macro de 0,58 do modelo Few-Shot aponta limitações de generalização entre as classes, exigindo estratégias complementares, como aumento de dados ou ajuste fino, para a sua aplicação em ambientes de produção.

4. Conclusões e perspectivas futuras

Os resultados deste trabalho evidenciam que os LLMs constituem uma alternativa viável e promissora para a predição do tempo de execução de *jobs* em ambientes de HPC. Diferentemente de abordagens tradicionais de estimativa baseadas em heurísticas estáticas ou modelos de aprendizado de máquina, a utilização de LLMs permite interpretar diretamente o conteúdo dos scripts de submissão, incluindo parâmetros de simulação, caminhos de arquivos e metadados, extraindo padrões semânticos que seriam difíceis de capturar por métodos convencionais. Essa capacidade de compreensão contextual representa um avanço, pois possibilita que estimativas de tempo sejam geradas sem a necessidade de executar previamente o *job* ou de manter modelos preditivos treinados especificamente para cada tipo de simulador. Dessa forma, a abordagem proposta tem potencial para ser integrada a fluxos de trabalho de submissão de *jobs*, oferecendo aos usuários e administradores de sistemas HPC uma ferramenta de apoio à decisão que contribua para o planejamento mais eficiente de filas, a alocação otimizada de recursos computacionais e a redução do desperdício de horas de processamento.

Os resultados obtidos abrem caminhos promissores para a aplicação de LLMs na otimização de ambientes HPC. Uma possível aplicação é seu uso na construção de assistentes conversacionais especializados (*chatbots*) para recomendações de *tuning* de scripts de submissão. Com base na capacidade demonstrada do modelo em interpretar scripts e correlacioná-los a faixas de tempo de execução, é possível conceber um

sistema interativo que analise o script do usuário e forneça sugestões concretas de otimização, como ajuste do número de processadores, redimensionamento de blocos ativos, seleção mais adequada de simulador ou reestruturação de parâmetros de entrada, visando reduzir o tempo de execução e, consequentemente, o consumo de recursos computacionais. Adicionalmente, pretende-se explorar técnicas de aumento de dados e amostragem estratificada para mitigar o impacto do desbalanceamento de classes, bem como avaliar abordagens de ajuste fino (fine-tuning) do modelo em dados específicos do domínio. A integração dessas predições com sistemas de escalonamento de filas (como Slurm) também representa uma perspectiva relevante, permitindo que estimativas de tempo de execução geradas por LLMs alimentem diretamente as políticas de alocação de recursos, contribuindo para maior eficiência operacional do ambiente HPC.

References

Chase, H. (2022). LangChain [Computer software]. GitHub. https://github.com/langchain-ai/langchain

Liu, H., Ma, et al. (2025). ORA: Job Runtime Prediction for High-Performance Computing Platforms Using the Online Retrieval-Augmented Language Model. In *Proceedings of the 39th ACM International Conference on Supercomputing*, pages 884–894.

Meng, X., et al. (2025). A Review of Parallel Computing for Large-scale Reservoir Numerical Simulation. *Archives of Computational Methods in Engineering*, 32(7):4125–4162.

Middya, U., Manea, A., Alhubail, M., Ferguson, T., Byer, T., and Dogru, A. (2021). A massively parallel reservoir simulator on the GPU architecture. In *SPE reservoir simulation conference*, page D011S010R002. SPE.

Nunes, A. L., et al. (2025a). Two-Step Estimation Strategy for Predicting Petroleum Reservoir Simulation Jobs Runtime on an HPC Cluster. *Concurrency and Computation: Practice and Experience*, 37(4-5):e70026.

Nunes, A. L., et al. (2025b). Explorando Modelos Preditivos de Tempo de Execução de Simulações de Reservatório em Clusters HPC. In *Escola Regional de Alto Desempenho da Região Sudeste (ERAD-SE)*, páginas 6–10. SBC.

Portella, F., et al. (2022). TunaOil: A tuning algorithm strategy for reservoir simulation workloads. *Journal of Computational Science*, 63:101783.

Tuczyński, T. and Stopa, J. (2023). Uncertainty quantification in reservoir simulation using modern data assimilation algorithm. *Energies*, 16(3):1153.

Wang, H., Dai, Y. Q., Yu, J., and Dong, Y. (2021). Predicting running time of aerodynamic jobs in HPC system by combining supervised and unsupervised learning method. *Advances in Aerodynamics*, 3(1):22.