Dataset for Validation and Performance Testing of Online Judges

Dataset para Validação e Teste de Desempenho de Julgadores Automáticos de Código

Bruno Ravi Mendes do Vale[1], **Renato Cavalcante Chaves**[1], **Kerven Maciel Monteiro de Albuquerque**[1], **Roberto Carlos de Souza**[1], **Thiago Gouveia**[1]

[1]Instituto Federal da Paraíba (IFPB)
João Pessoa – Paraíba – Brasil

bruno.ravi25@gmail.com

Abstract. *This study proposes a dataset generated from real coding competitions, intended for the validation and performance testing of Online Judges (OJ). The proposed dataset is employed to assess an OJ developed at a higher education institution. The results obtained indicate the successful achievement of the intended goal, in addition to providing a baseline that can be utilized as a reference for comparing with other OJs or for validating improvements between different versions of the same OJ.*

Keywords*: Online Judges, Benchmark, Performance Testing.*

Resumo. *Este trabalho propõe um conjunto de dados gerado a partir de competições reais, destinado à validação e teste de desempenho de Julgadores Automáticos de Código (JAs). O dataset proposto é utilizado para avaliar um JA desenvolvido em uma instituição de ensino superior. Os resultados obtidos demonstram que o objetivo proposto foi atendido, além de oferecer uma linha de base que pode ser usada como parâmetro de comparação com outros JA, ou para validar melhorias entre versões de um mesmo JA.*

Palavras-chave*: Juízes Online, Benchmark, Testes de Performance.*

1. Introdução

Não se pode negar que a Internet tem se tornado, dia pós dia, mais onipresente na sociedade, em todas as suas dimensões. Em 2022, aproximadamente 5,3 bilhões de pessoas usavam a Internet, o que representa quase dois terços da população mundial [International Telecommunication Union 2022]. Para muitas pessoas, a Internet tornou-se um instrumento essencial para estudos, trabalho, entretenimento e comunicação com amigos e familiares, além de impulsionar a pesquisa e a inovação tecnológica, tomando como base um mundo conectado no qual a troca de informações ocorre a qualquer hora e em tempo real. Tal disponibilidade de informações aliada ao decorrente avanço tecnológico, especialmente na área de Computação, tornou a Programação de Computadores uma habilidade importante tanto para a academia quanto para a indústria. Neste sentido, vários países desenvolvidos têm dado ênfase ao ensino de Programação na educação básica, inclusive tornando-a disciplina curricular obrigatória [Watanobe et al. 2020a].

No entanto, programar não pode ser considerado uma tarefa fácil, sendo comum que estudantes enfrentem diversos obstáculos durante o seu aprendizado. Dentre os estudos que tratam desse problema, [Souza et al. 2016] realizam um mapeamento sistemático e identificam algumas causas como dificuldades de aprendizagem de conceitos ou aplicação destes no código. Algumas estratégias utilizadas para mitigação dessas dificuldades são a colaboração com colegas, *feedbacks* significativos, desafios baseados em problemas reais e ambientes de desenvolvimento pedagógico, com funcionalidades para construção e execução de programas, como é o caso dos Juízes Online (OJs, do inglês *Online Judges*).

[Wasik et al. 2018a] definem OJs como sistemas confiáveis e seguros que objetivam avaliar códigos submetidos pelos usuários. Eles disponibilizam problemas envolvendo algoritmos, estruturas de dados e lógica de programação, e permitem que estudantes submetam suas soluções para serem avaliadas e receberem um *feedback* imediato. Também é comum em OJs a realização de competições de programação, nas quais um conjunto de problemas é disponibilizado para todos os competidores, que devem resolvê-los, individualmente ou em times, na maior quantidade possível e no menor tempo. Em seu núcleo, OJs dependem de julgadores automáticos (JA) para, de fato, compilar, executar e avaliar as soluções submetidas. Embora várias pesquisas tratem diretamente dos OJs e suas aplicações, não encontram-se muitas ferramentas para validar e aferir o desempenho de julgadores automáticos.

Neste sentido, este trabalho propõe um *dataset*, gerado a partir de competições reais, para ser usado como modelo para validação e testes de desempenho de JAs. Adicionalmente, com objetivo de oferecer uma linha de referência (*baseline*), o *dataset* desenvolvido é utilizado para validar e avaliar o desempenho de um julgador automático desenvolvido e utilizado em uma instituição de ensino superior. Para melhor descrever o estudo realizado, o restante deste artigo está organizado como segue: a Seção 2 aborda alguns trabalhos relacionados ao tema, apresentando uma revisão da literatura relevante para o assunto em discussão; a Seção 3 apresenta algumas Competições de Programação e sua devida importância no contexto desta pesquisa; a Seção 4 apresenta detalhes sobre o *dataset* desenvolvido neste estudo, destacando seu formato e suas características; a Seção 5 descreve os experimentos computacionais realizados e apresenta uma análise sucinta sobre os resultados e suas implicações. Por fim, na Seção 6, oferecemos as Considerações Finais, resumindo os principais resultados, contribuições e possíveis direções para pesquisas futuras.

2. Trabalhos relacionados

Esta seção aborda algumas pesquisas relacionadas ao tema deste trabalho. Inicialmente são apresentados os Juízes Online mais conhecidos. Em sequência, são examinamos estudos que estabelecem conexões entre os Juízes Online e o processo de ensino/aprendizagem de programação, assim como é realizada uma uma breve revisão da literatura relacionada à arquitetura e à construção dessas plataformas.

Juízes Online são ambientes de aprendizagem, comumente disponibilizados na Internet, destinados a estudantes de Computação, oferecendo uma diversidade de exercícios de programação e tarefas relacionadas a conceitos desta disciplina. Estas plataformas desempenham um papel crucial na orientação dos estudantes para a aplicação de seus

conhecimentos teóricos na resolução de problemas práticos de programação, contribuindo, desse modo, para o aprimoramento de suas habilidades e competência nesse domínio [Wang et al. 2023]. Dentre os OJs mais conhecidos, podem ser citados o Hackerrank [Sreeram et al. 2018], o UVa Online Judge [Revilla et al. 2008], o CodeForces [Mirzayanov et al. 2020], o CodeBench [Galvão et al. 2016] e o Beecrowd, antes chamado de URI Online Judge [Tonin et al. 2012]. [Wasik et al. 2018b] oferecem um *survey* sobre OJs e suas aplicações.

De acordo com [Mani et al. 2014], a grande maioria dos JOs fornece apenas vereditos binários, classificando as respostas submetidas como ”suficientemente boas”ou não. Tal abordagem é apropriada para plataformas competitivas ou de recrutamento, mas pode ser limitante em ambientes educacionais, onde é desejável que se forneça *feedbacks* mais detalhados. Neste sentido, o trabalho propõe uma abordagem baseada na mineração de dados de submissões incorretas passadas para extrair conjuntos de testes mais relevantes. [Francisco and Ambrosio 2015] apresentam a especificação preliminar da arquitetura e os requisitos funcionais do PROBOCA, uma ferramenta de aprendizado de programação com base no Juiz Online BOCA [de Campos and Ferreira 2004]. O sistema oferece um banco de dados de problemas classificados por tema e nível de dificuldade, juntamente com de funcionalidades de apoio ao processo de ensino, destacando-se a estimativa de dificuldade de problemas, que quando comparada à percepção dos alunos, permite a identificação de estudantes com dificuldades.

Segundo [Hidalgo-Céspedes et al. 2020], JOs são ferramentas muito úteis e por vezes indispensáveis para apoiar a programação competitiva, o recrutamento de profissionais e o ensino de programação. Os autores destacam a negligência das necessidades dos professores, que desempenham um papel central na adoção de ferramentas educacionais. O estudo coleta 132 requisitos funcionais para JOs educacionais, categorizando-os em 27 grupos e verifica seu atendimento em quatro JO educacionais. [Došilović and Mekterović 2020] apresentam o **Judge0**, descrito pelos autores como um sistema de execução de código *online* inovador, robusto, escalável, de código aberto, e que possui uma arquitetura modular moderna que pode ser implantada em diferentes sistemas operacionais. O trabalho discute o *design*, e os desafios enfrentados na construção de tais sistemas, comparando com outras soluções e possíveis casos de uso, desde plataformas de programação competitiva até ambientes educacionais e de recrutamento.

[Watanobe et al. 2020b] apontam que um JO pode funcionar como plataforma educacional de programação, fazendo uso de atividades diárias para gerar objetos de aprendizagem, como tarefas, códigos de solução, ou avaliações. Neste contexto, agentes inteligentes de *software* podem criar um ecossistema com esses objetos. O artigo propõe uma arquitetura que implementa esse ecossistema com base em um JO, abordando benefícios potenciais e desafios de pesquisa. Vale observar que [Hidalgo-Céspedes et al. 2021] argumentam que a maioria dos JOs foi concebida inicialmente para competições, deixando de atender plenamente às necessidades do ensino de programação. Neste contexto, apresentam um design de interface de usuário para um JO enriquecido com recursos de um sistema de gerenciamento de aprendizagem para atender aos requisitos de professores e alunos em programação concorrente. Um protótipo foi avaliado por estudantes com experiência em JO, que conseguiram concluir as tarefas propostas e expressaram alta satisfação com o uso da ferramenta.

[Wang et al. 2021] apresentam o **MetaOJ**, um JO que permite a realização de testes de programação em larga escala. O MetaOJ foi projetado para transformar um JO comum em um sistema distribuído, tolerante a falhas e escalável, adicionando interfaces e criando múltiplas instâncias. A modificação acrescenta menos de 3% de linhas de código e tem uma perda de desempenho de menos de 12% para instâncias únicas. O MetaOJ oferece integração com infraestruturas em nuvem para automação do processo de implantação. Desenvolver e operar um Sistema de Juiz Online (OJS) para avaliar programas é desafiador devido a diversas necessidades funcionais e não funcionais. Embora muitos OJSs tenham sido criados e usados com sucesso, a teoria por trás da construção de OJSs não foi amplamente discutida. Neste artigo, apresentamos os requisitos funcionais e não funcionais para OJS, detalhamos a arquitetura de um OJS em operação há mais de uma década e compartilhamos experiências e desafios do mundo real enfrentados durante essa jornada.

[Watanobe et al. 2022] discutem que desenvolver e operar um JO para avaliar programas é desafiador devido aos seus diversos requisitos funcionais e não funcionais. Embora muitos JOs tenham sido criados e usados com sucesso, a teoria por trás da construção de OJSs não foi amplamente discutida. Os autores apresentam requisitos funcionais e não funcionais para JOs, detalham a arquitetura de um JO em operação há mais de uma década e compartilham experiências e desafios do mundo real enfrentados durante essa jornada. Por fim, [Gouveia et al. 2023] descrevem a arquitetura do **C073**, uma ferramenta que apoia o uso da metodologia ABP (Aprendizagem Baseada em Problemas) em disciplinas de programação. O C073 destaca-se pelo módulo de cursos, onde professores podem adicionar conteúdo e acompanhar o desempenho dos alunos. Foi projetado para apoiar a ABP, oferecendo soluções eficazes para desafios comuns no ensino de Programação.

3. Composição do Dataset

Para gerar o *dataset* proposto, foram utilizadas as submissões da Maratona POP Médio 2022, da Maratona POP Superior 2022 e do Aquecimento comum a ambas [Ditarso et al. 2022]. Também foi utilizada foi lista de submissões da primeira fase da Maratona SBC de Programação 2021 [Sociedade Brasileira de Computação 2021]. Nesse caso, os códigos não estão disponíveis publicamente, então foram considerados apenas o usuário que enviou a submissão, o tempo e o veredito.

Além de coletar dados de diferentes competições, o dataset em si é estruturado como várias competições independentes, que podem ser utilizadas para os testes de JAs tanto isoladamente, quanto em conjunto. A existência de múltiplas competições permite avaliar diferentes aspectos, como duração e distribuição das submissões ao longo do tempo. Esses aspectos são detalhados nas próximas subseções, para cada uma das competições consideradas.

3.1. Maratona POP

A Maratona POP Médio de 2022 foi realizada no dia 16 de setembro de 2022, em times de até dois integrantes, que receberam um caderno com 10 problemas inéditos e deveriam resolver o maior número possível dentro do limite de 3 horas. Essa competição foi exclusiva para estudantes regularmente matriculados em algum curso de nível médio de um dos *campi* do IFPB. Por sua vez, a Maratona POP Superior de 2022 foi realizada em 24

de setembro de 2022, em times de até três integrantes, com 11 problemas e duração de 3 horas. Os competidores foram estudantes regularmente matriculados em algum curso de nível superior de um dos *campi* do IFPB. Esta modalidade foi dividida nas categorias Iniciante (até o terceiro período) e Avançado (após o terceiro período). Ambas as competições ocorreram presencialmente no *campus* João Pessoa do Instituto Federal da Paraíba (IFPB), como desafios acadêmicos integrantes da XVII Semana de Educação, Ciência, Cultura e Tecnologia (SECT 2022), com premiações para os melhores times [Instituto Federal da Paraíba 2022].

Ao compor a base para o *dataset*, a Maratona POP Médio e as duas modalidades da Maratona POP Superior foram agrupadas em uma só competição (que será chamada Maratona POP). Os principais fatores na geração de competições para o *dataset* são a duração da competição e quantidade de competidores. Sendo assim, esse agrupamento torna-se a base para novas competições de 3 horas.

A Figura 1 apresenta o histograma das submissões da Maratona POP, em intervalos de 10 min. Observa-se que o volume de submissões é maior no início da competição e decresce nas primeiras 2 h de prova. Nos últimos 30 min, a quantidade de submissões começa a subir levemente, mas permanece num nível bem inferior ao começo da prova.

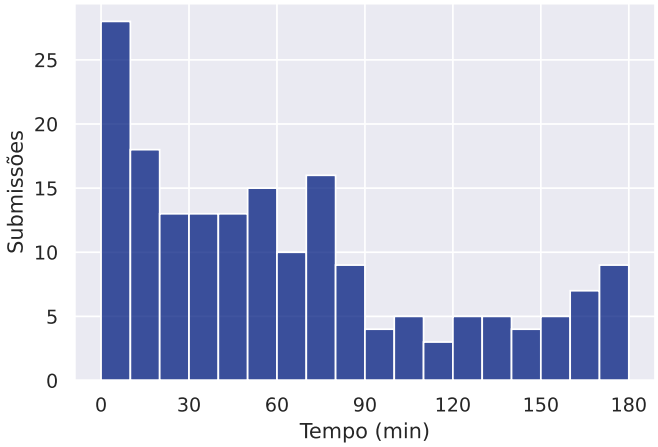

Figura 1. Histograma de submissões da Maratona POP

3.2. Aquecimento

O Aquecimento das Maratonas POP Médio e Superior foi realizado no dia 10 de setembro de 2022, no mesmo formato das competições. Neste caso, puderam participar os times inscritos em umas das modalidades da Maratona POP, e a prova consistiu em 6 problemas de edições anteriores e duração de 1 hora. Ao contrário das competições principais, o objetivo do Aquecimento era familiarizar os competidores com a plataforma e testar o ambiente de execução, de modo que não houve restrições quanto à comunicação entre participantes, auxílio dos colegas e professores, e consulta à Internet, por exemplo.

A Figura 2 apresenta o histograma das submissões do Aquecimento, em intervalos de 5 min. O volume de submissões, neste caso, varia ao longo da competição, mas permanece alto na maior parte do tempo. Mesmo nos momentos com menos submissões, não fica abaixo da metade das submissões dos momentos mais intensos.

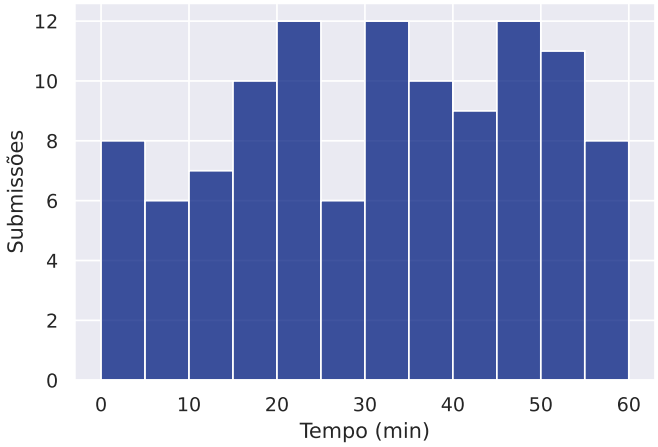

Figura 2. Histograma de submissões do Aquecimento da Maratona POP

3.3. Maratona SBC

Outra competição utilizada foi a primeira fase da Maratona SBC de Programação de 2021 [Sociedade Brasileira de Computação 2021]. A Maratona SBC ocorre em duas fases, como parte da regional sul-americana do International Collegiate Programming Contest [ICPC Foundation 2023]. É destinada a alunos de graduação e início de pós-graduação, que competem em times formados por três integrantes, com 5 h de prova em cada uma das fases.

A Maratona SBC não disponibiliza os códigos submetidos publicamente, então foram considerados apenas o usuário que enviou a submissão, o tempo e o veredito (resultado dado pelo OJ). Para substituir os códigos ausentes, utilizou-se submissões das demais competições, buscando opções com veredito similar. Para determinar os vereditos similares, estes foram agrupados em três categorias: submissões aceitas, submissões incorretas por estourarem o tempo limite, e demais submissões incorretas.

A Figura 3 apresenta o histograma das submissões da Maratona SBC, em intervalos de 10 min. O início da competição apresenta uma quantidade pequena de submissões. Em contraste, há um pico de submissões no final. No resto da competição, há pouca variação no volume de submissões, que permanece num nível intermediário entre os volumes do início e do final.

4. Dataset

Esta seção descreve o dataset proposto para validação e verificação de desempenho de julgadores online. Foram definidas três durações (1 h, 3 h e 5 h) e oito configurações de quantidade de competidores (20, 50, 100, 200, 500, 1000, 200 e 5000 usuários), totalizando 24 competições. Uma competição C é formada por um conjunto P de problemas, um conjunto de competidores U e outro de submissões S. O tempo de cada submissão é um valor entre 0 e a duração da competição t_C, excluídos os extremos. Sendo assim, o procedimento para geração de uma nova competição tem como parâmetros a duração e a quantidade de competidores, além de pelo menos uma competição de referência. Então são gerados novos competidores, até a quantidade determinada, a partir das submissões das competições de referência.

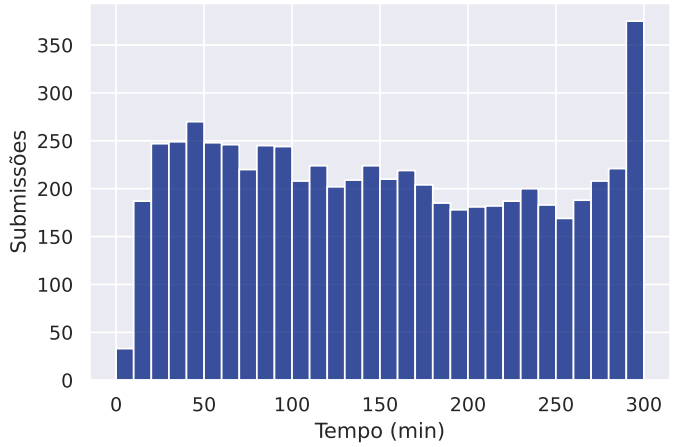

Figura 3. Histograma de submissões da Maratona SBC de Programação 2021

Para cada novo usuário, determina-se aleatoriamente uma quantidade q de problemas resolvidos. O algoritmo não força que o usuário tenha resolvido exatamente a quantidade preliminar de problemas, mas constrói as submissões do usuário com base nos usuários reais que resolveram um número similar de problemas. Para manter o padrão observado nas competições, as probabilidades são ponderadas pelo *score* geral. Ou seja, se n_p usuários resolveram p problemas, então a probabilidade de cada p ser escolhido será de n_p/N, onde $N = \sum n_p$. Define-se Q como o conjunto de usuários da competição de referência que resolveram entre $q - 1$ e $q + 1$ problemas, inclusive.

Em relação ao tempo, a competição é dividida em intervalos t_i, $i = 1, \ldots, n$ sucessivos, de duração constante e igual a 5 min. Em cada intervalo, escolhe-se aleatoriamente um usuário $u \in Q$ e suas submissões são atribuídas ao usuário que está sendo criado. Os tempos de cada submissão são modificados aleatoriamente, para evitar que as submissões estejam concentradas em instantes específicos. A modificação dos tempos respeita o intervalo em questão e a ordem original das submissões.

Para gerar o dataset, foi utilizada a linguagem Python e cada competição consiste num arquivo CSV, contendo informações sobre as submissões. Cada linha do arquivo representa uma submissão, informando o tempo em milisegundos do início da competição, qual competidor submeteu, para qual problema e a categoria do veredito. Além disso, também é informado o nome de um modelo de submissão que pode ser utilizado ao simular aquela competição em algum julgador. A Figura 4 exibe a estrutura do arquivo que descreve a competição com duração de 1 hora e 20 usuários. Os arquivos das competições, assim como o conjunto de problemas e submissões estão disponíveis publicamente em `https://tinyurl.com/dataset-contests`.

4.1. Histogramas de submissões

Como apresentado na Seção 3, cada competição de referência possui características específicas, que são evidentes nos histogramas da quantidade de submissões em cada momento da prova. Buscou-se manter essas características nas competições geradas, como está ilustrado nas Figuras 5–10.

As Figuras 5 e 6 mostram os histogramas de dois exemplos de competições com 1 h de duração. Assim como o Aquecimento utilizado como referência, o volume de sub-

	time	user	problem	submission	veredict
0	165	BV61	1U	1US	2
1	181	NI74	1U	1US	2
2	200	LC87	1V	1VK	2
3	208	BV61	1U	1UK	0
4	218	NI74	1U	1UK	0
...	...	...	...	...	...
176	3592	JE13	1Y	1YO	2
177	3593	RR18	1Z	1ZX	2
178	3593	WK79	1Y	1YO	2
179	3595	BV61	1Y	1YO	2
180	3598	XW76	1Z	1ZX	2

181 rows × 5 columns

Figura 4. Estrutura do arquivo CSV da competição com 1 h e 20 competidores

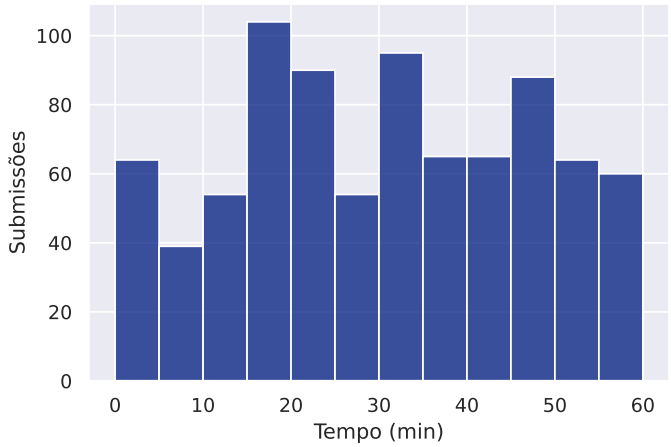

Figura 5. Histograma de submissões da competição gerada com 1 h e 100 competidores

missões tem alguma variação, mas permanece sempre alto. Os gráficos não são idênticos ao da Figura 2 devido à aleatoriedade empregada no processo de geração, mas é possível observar semelhanças, como o menor volume no meio e nos extremos e os picos aos 20 min e aos 35 min.

As Figuras 7 e 8 são exemplos de competições com 3 h de duração. Elas usaram a Maratona POP como referência, logo apresentam semelhança no pico de submissões no início da competição, seguido por um número cada vez menor de submissões, voltando a subir levemente no final. As Figuras 9 e 10 exemplificam competições com 5 h de duração. Elas usaram a Maratona SBC como referência, caracterizada pelo baixo volume de submissões no início e um pico maior no final. Uma variação maior que o padrão da Maratona SBC ocorreu na Figura 9, gerando um pico de valor próximo ao máximo da competição. Um fator para explicar esse comportamento é a quantidade de competidores

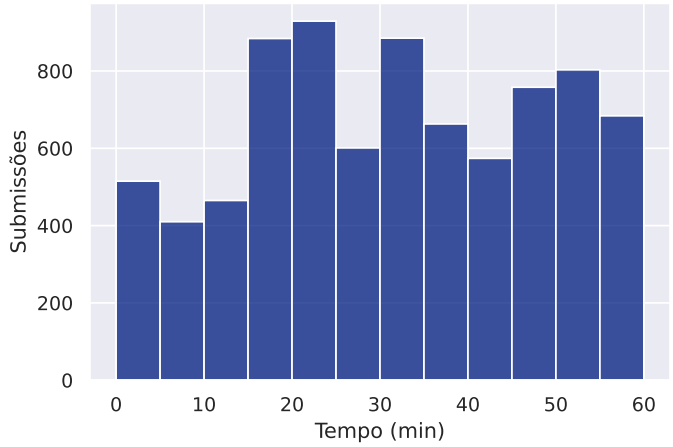

Figura 6. Histograma de submissões da competição gerada com 1 h e 1000 competidores

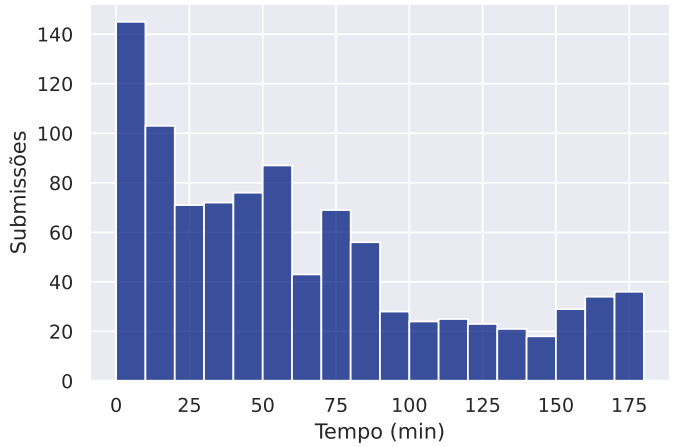

Figura 7. Histograma de submissões da competição gerada com 3 h e 100 competidores

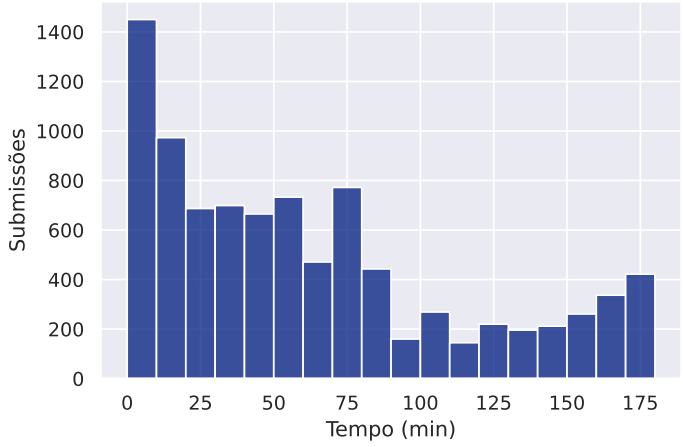

Figura 8. Histograma de submissões da competição gerada com 3 h e 1000 competidores

(100) inferior à da competição de referência (492). Neste caso, a aleatoriedade torna-se mais visível, ofuscando as características originais da competição.

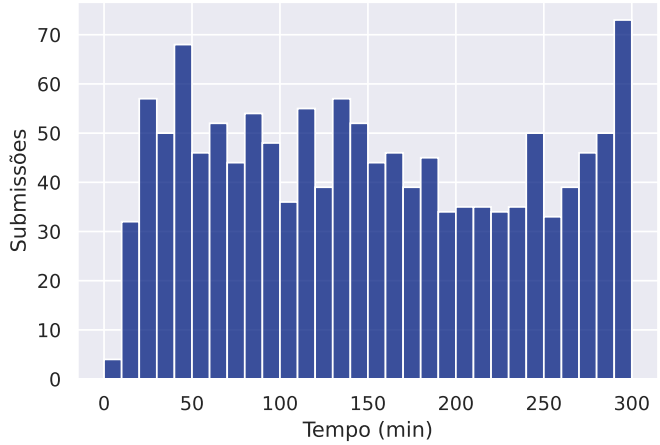

Figura 9. Histograma de submissões da competição gerada com 5 h e 100 competidores

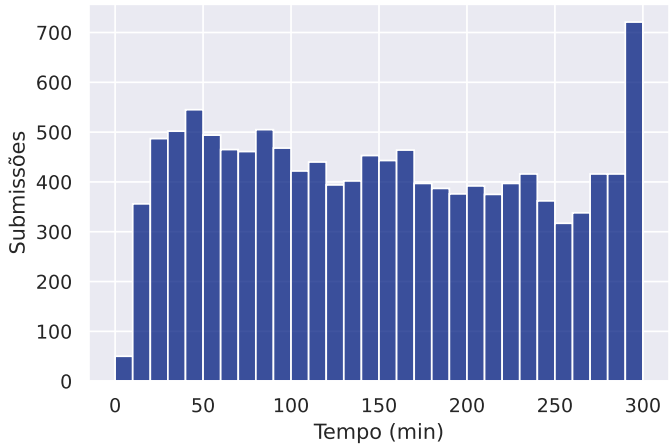

Figura 10. Histograma de submissões da competição gerada com 5 h e 1000 competidores

4.2. Experimentos

Com objetivo de validar o *dataset* desenvolvido e oferecer um resultado para ser usado como referência, todas as competições foram simuladas na ferramenta C073 [Gouveia et al. 2023]. A simulação foi realizada em um computador com 16 GB de RAM DDR4, processador Intel Core i7-7500U (64 bits), com CPU contendo 2 núcleos de 2,70 GHz, e executando o sistema operacional Ubuntu 22.10. Cada competição foi executada até todas as submissões serem enviadas e julgadas, de modo que não reste nenhuma submissão na fila. Os resultados estão apresentados na Tabela 1.

5. Considerações finais

Este trabalho propõe um *dataset*, gerado a partir de dados de competições reais, para ser usado como modelo para validação e testes de desempenho de JAs. Adicionalmente,

Tabela 1. Testes com o *dataset* no N30

t_C	$\|U\|$	Tempo de fila (s)			Tempo de execução (s)		
		Mínimo	Médio	Máximo	Mínimo	Médio	Máximo
1 h	20	0.12	2.80	15.52	0.83	3.21	12.42
	50	0.07	25.89	183.67	0.87	3.12	13.72
	100	0.10	47.40	254.77	0.79	3.14	13.68
	200	0.75	905.82	2314.64	0.78	3.14	13.96
	500	2.16	4746.58	10688.09	0.79	3.18	13.80
	1000	0.17	10963.53	23956.79	0.77	3.14	14.15
	2000	3.01	24364.50	51903.84	0.73	3.16	14.02
	5000	0.06	69757.93	146063.01	0.80	3.29	14.25
3 h	20	0.11	1.94	11.19	0.82	3.11	7.08
	50	0.07	2.62	17.09	0.77	2.98	7.12
	100	0.06	7.86	58.16	0.81	2.97	8.25
	200	0.11	99.09	460.67	0.80	3.06	7.41
	500	1.39	3112.03	5577.43	0.75	3.08	7.52
	1000	1.16	9855.31	17787.86	0.77	3.04	7.57
	2000	1.37	24436.99	48182.57	0.76	3.16	7.64
	5000	1.87	75026.30	150321.49	0.79	3.28	19.24
5 h	20	0.08	1.74	7.03	0.84	3.24	6.81
	50	0.06	1.89	14.50	0.78	3.15	6.65
	100	0.07	2.56	21.28	0.78	3.20	6.94
	200	0.06	6.57	124.83	0.77	3.15	6.93
	500	0.18	2044.38	3971.63	0.75	3.19	8.28
	1000	0.08	11793.79	23664.80	0.76	3.16	7.81
	2000	0.14	35274.55	70361.44	0.74	3.24	7.68
	5000	0.08	112477.53	222950.97	0.87	3.41	9.97

com objetivo de oferecer um ponto de comparação (uma linha de referência, ou *baseline*), o conjunto de dados desenvolvido é utilizado para validar e avaliar o desempenho de um julgador automático desenvolvido e utilizado em uma instituição de ensino superior. Os experimentos computacionais realizados atestaram a eficácia do *benchmark* proposto, possibilitando que as funcionalidades básicas do JO fossem validadas com sucesso. Em adição, há também uma linha base para melhoria e comparação de desempenho entre JOs.

Como trabalhos futuros, pretendemos estender o *dataset* para que este permita, também, que sejam realizados testes de segurança nos JOs, visto que estes lidam com dados de fontes pouco confiáveis, e comumente sofrem ataques que visam sobrecarregar sua memória, processamento ou disco.

Agradecimentos

Os autores agradecem ao PIBITI/CNPq, POP e IFPB pelo financiamento parcial desta pesquisa.

Referências

de Campos, C. P. and Ferreira, C. E. (2004). Boca: um sistema de apoio a competições de programação. In *Workshop de Educação em Computação*. Sociedade Brasileira de Computação.

Ditarso, P., Gouveia, T., and Cavalcanti, V. (2022). POP: Projeto Olímpico de Programação. Disponível em `https://joaopessoa.ifpb.edu.br/pop/`.

Došilović, H. Z. and Mekterović, I. (2020). Robust and scalable online code execution system. In *2020 43rd International Convention on Information, Communication and Electronic Technology (MIPRO)*, pages 1627–1632. IEEE.

Francisco, R. E. and Ambrosio, A. P. (2015). Mining an online judge system to support introductory computer programming teaching. In *EDM (Workshops)*. Citeseer.

Galvão, L., Fernandes, D., and Gadelha, B. (2016). Juiz online como ferramenta de apoio a uma metodologia de ensino híbrido em programação. In *Brazilian Symposium on Computers in Education (Simpósio Brasileiro de Informática na Educação-SBIE)*, volume 27, page 140.

Gouveia, T., Albuquerque, K. M. M., Oliveira, J. D., and Maciel, V. M. B. C. (2023). C073: Ferramenta para apoio ao ensino de programação usando a metodologia de aprendizagem baseada em problemas. *Revista Principia - Divulgação Científica e Tecnológica do IFPB*, 60(1):70–87.

Hidalgo-Céspedes, J., Marín-Raventós, G., and Calderón-Campos, M. E. (2020). Online judge support for programming teaching. In *2020 XLVI Latin American Computing Conference (CLEI)*, pages 522–530. IEEE.

Hidalgo-Céspedes, J., Marín-Raventós, G., and Calderón-Campos, M. E. (2021). Usability of an online judge for concurrent programming education. In *2021 XVI Latin American Conference on Learning Technologies (LACLO)*, pages 318–325. IEEE.

ICPC Foundation (2023). The ICPC International Collegiate Programming Contest. Disponível em `https://icpc.global/`.

Instituto Federal da Paraíba (2022). XVII Semana de Educação, Ciência, Cultura e Tecnologia. Disponível em `https://www.even3.com.br/sect2022/`.

International Telecommunication Union (2022). Measuring digital development: Facts and figures 2022. Disponível em `https://www.itu.int/itu-d/reports/statistics/facts-figures-2022/`.

Mani, A., Venkataramani, D., Petit Silvestre, J., and Roura Ferret, S. (2014). Better feedback for educational online judges. In *Proceedings of the 6th International Conference on Computer Supported Education, Volume 2: Barcelona, Spain, 1-3 April, 2014*, pages 176–183. SciTePress.

Mirzayanov, M., Pavlova, O., Mavrin, P., Melnikov, R., Plotnikov, A., Parfenov, V., and Stankevich, A. (2020). Codeforces as an educational platform for learning programming in digitalization. *Olympiads in Informatics*, 14:133–142.

Revilla, M. A., Manzoor, S., and Liu, R. (2008). Competitive learning in informatics: The UVa Online Judge experience. *Olympiads in Informatics*, 2(10):131–148.

Sociedade Brasileira de Computação (2021). XXVI Maratona de Programação - 2021. Disponível em https://maratona.sbc.org.br/hist/2021/result21.html.

Souza, D. M., da Silva Batista, M. H., and Barbosa, E. F. (2016). Problemas e dificuldades no ensino de programação: Um mapeamento sistemático. *Revista Brasileira de Informática na Educação*, 24(1):39.

Sreeram, N., Kumar, V. U., and Rao, L. S. (2018). A survey paper on modern online cloud-based programming platforms. *International Journal of Engineering & Technology*, 7(2.32):352–354.

Tonin, N. A., Zanin, F. A., and Bez, J. L. (2012). Enhancing traditional algorithms classes using URI Online Judge. In *2012 International Conference on E-Learning and E-Technologies in Education (ICEEE)*, pages 110–113. IEEE.

Wang, J., Lin, P., Tang, Z., and Chen, S. (2023). How problem difficulty and order influence programming education outcomes in online judge systems. *Heliyon*.

Wang, M., Han, W., and Chen, W. (2021). Metaoj: A massive distributed online judge system. *Tsinghua Science and Technology*, 26(4):548–557.

Wasik, S., Antczak, M., Badura, J., Laskowski, A., and Sternal, T. (2018a). A survey on online judge systems and their applications. *ACM Computing Surveys (CSUR)*, 51(1):1–34.

Wasik, S., Antczak, M., Badura, J., Laskowski, A., and Sternal, T. (2018b). A survey on online judge systems and their applications. *ACM Computing Surveys (CSUR)*, 51(1):1–34.

Watanobe, Y., Intisar, C., Cortez, R., and Vazhenin, A. (2020a). Next-generation programming learning platform: Architecture and challenges. In *SHS Web of Conferences*, volume 77, page 01004. EDP Sciences.

Watanobe, Y., Intisar, C., Cortez, R., and Vazhenin, A. (2020b). Next-generation programming learning platform: Architecture and challenges. In *SHS Web of Conferences*, volume 77, page 01004. EDP Sciences.

Watanobe, Y., Rahman, M. M., Matsumoto, T., Rage, U. K., and Ravikumar, P. (2022). Online judge system: Requirements, architecture, and experiences. *International Journal of Software Engineering and Knowledge Engineering*, 32(06):917–946.