IMPLEMENTAÇÃO E COMPARAÇÃO DE ALGORITMOS PARA ESCALONAMENTO DE TAREFAS EM HPC[1]

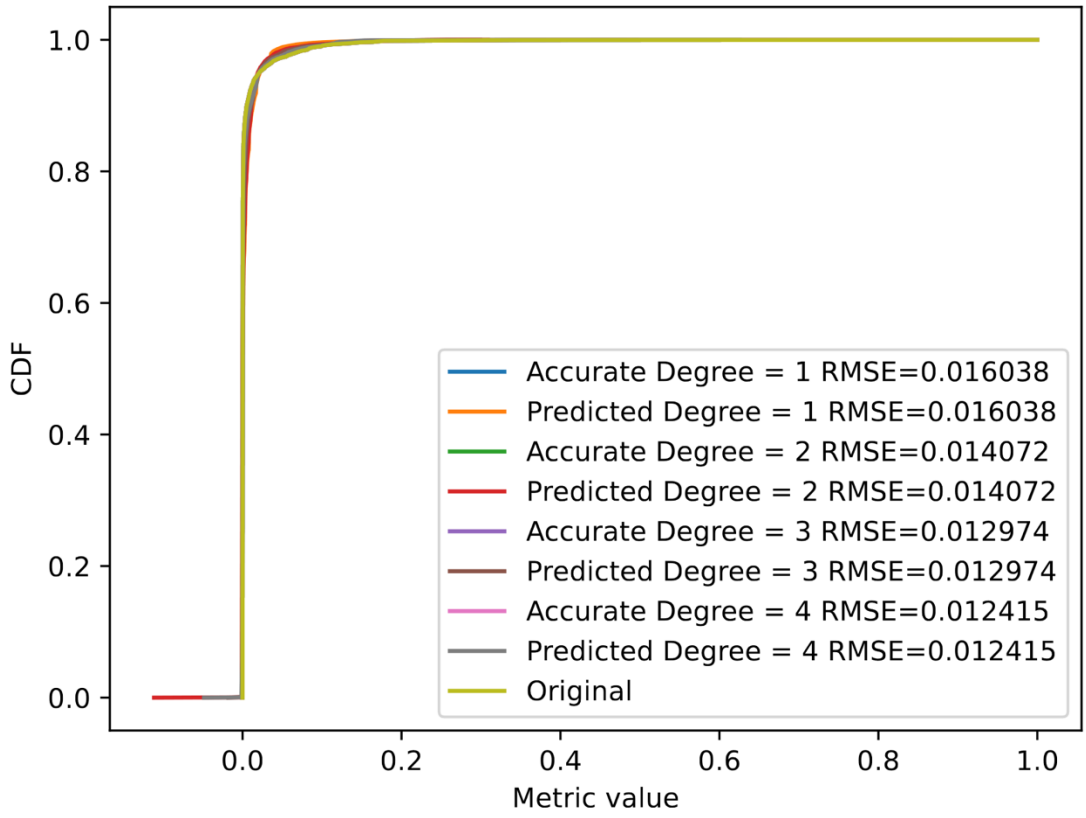
Ana Eloina Nascimento Kraus[12], Guilherme Piêgas Koslovski[13]

[1] Vinculado ao projeto "Mecanismos para Alocação de Infraestruturas Virtuais baseados em Aprendizado de Máquina e Acelerados por GPUs"
[2]Acadêmico (a) do Curso de Ciência da Computação – CCT – Bolsista PIBIC/CNPq
[3] Orientador, Departamento de Ciência da Computação – CCT – guilherme.koslovski@udesc.br

Escalonamento de tarefas é um problema de negócios reais e virtuais tanto quanto. Para esta pesquisa, considera-se o *workflow* de um sistema operacional multitarefa, que deve decidir quais processos priorizar entre centenas que chegam ao mesmo tempo, de tal maneira que todos tenham algum tempo de execução e preferencialmente terminem com êxito. O usuário, ainda que indiretamente, requisita um intervalo de tempo estimado para que suas tarefas/seus trabalhos executem, e tais intervalos são frequentemente imprecisos, apesar de importantes. No contexto do presente trabalho, os usuários submetem tarefas para escalonadores de infraestruturas *High Performance Computing* (HPC). Algoritmos atuais de escalonamento de tarefas empregam diferentes critérios, como: ordem de chegada; tempo solicitado; recursos solicitados; ou área total (ambos tempo e recursos), buscando ao máximo evitar um estado de *deadlock*, onde processos ficam mutuamente bloqueados, ou *starvation*, onde nunca é conferido a um processo específico algum tempo de CPU. O objetivo da pesquisa foi comparar algoritmos de escalonamento de tarefas, baseados no seguinte conhecimento prévio sobre: tempo estimado de execução, recursos solicitados, tempo de espera, máquinas computando e máquinas ociosas.

Para realizar a comparação é necessário saber quão bem os algoritmos performam em comparação ao *state-of-the-art*. No ambiente de desenvolvimento Nix, no qual todas as dependências necessárias são satisfeitas *out-of-the-box*, utilizou-se a *API* pybatsim, na linguagem de programação Python, a fim de simular o escalonamento de um *batch* de tarefas por uma coletânea de algoritmos de escalonamento já existentes. Testes foram executados com os algoritmos de escalonamento padrões e reconhecidos da literatura científica – além de algumas funções customizadas – alimentados com *workloads* paralelos reais de centros de computação como Intel, NASA, IBM e outros, para então calcular seus *scores* mistos de tempo e energia consumida por carga de trabalho na CPU. Em seguida, com os dados de *score* definitivos, foi possível trabalhar com regressão polinomial, CDF (*cumulative distribution function*) e *scatterplots* na análise da acurácia de um algoritmo que se baseie em 2, 4, e 5 grupos específicos de atributos de um *job*. Foram testados polinômios de graus 1 (linear) até 4, à procura da curva que melhor se encaixasse no (e logo, melhor previsse o) conjunto de dados em questão. Conjuntos X e Y de regressão múltipla polinomial foram divididos em 70% para treinamento e 30% para teste, além de serem normalizados e limpos (em relação a *outliers* estatísticos) para prepará-los para uso em inteligência artificial, dos quais foi possível extrair o modelo treinado mais próximo dos resultados da vida real e assim usá-lo na elaboração do novo algoritmo de escalonamento.

Figura 1. *Gráfico CDF demonstrando a curvatura de diversos polinômios em relação aos dados reais de consumo médio de energia.*



O CDF varia de 0 a 100% em Y e demonstra a quantidade dos *samples* de um *dataset* que é encontrada na faixa de valor respectiva em X. No gráfico acima, percebe-se que a maior parte dos *samples* no dataset tem os atributos de interesse com valores muito próximos do 0, e que 100% dos samples devem estar localizados, por exemplo, antes de 0.4, logo, com valores da métrica menores do que 0.4. Neste caso em específico, todos os modelos de regressão polinomial obtidos tiveram MSE (*mean squared error*, qualidade da predição; quanto menor o erro, melhor o modelo) baixo, e o escolhido entre eles é o de grau 4.

Palavras-chave: Escalonamento de tarefas. Regressão polinomial. Workloads paralelos.