

Escalonador *online* de *workflows* sensível ao estado da rede

Victor Tavares Alvarenga¹, Guilherme Piêgas Koslovski¹, Maurício Aronne Pillon¹

¹Programa de Pós-Graduação em Computação Aplicada - PPGCAP
Universidade do Estado de Santa Catarina - UDESC

Resumo. A execução de *workflows* distribuídos depende tanto dos recursos computacionais quanto da eficiência da comunicação entre tarefas. Políticas tradicionais de escalonamento priorizam precedência e custo computacional, desconsiderando o impacto do posicionamento das tarefas sobre a rede. Este artigo apresenta um escalonador online consciente da rede, agnóstico à política base, que ajusta a alocação considerando métricas topológicas e informações históricas. Um parâmetro controla o peso da intervenção da camada de rede. Avaliada sobre as heurísticas EASY, HEFT, CPOP e PEFT em uma topologia Dragonfly, a proposta reduziu em até 32% os saltos, 38% os fluxos entre racks, 51% os fluxos entre grupos e 33% o custo de comunicação, com significância estatística ($p < 0,05$).

Abstract. The execution of distributed *workflows* depends on both computational resources and the efficiency of communication among tasks. Traditional scheduling policies prioritize precedence constraints and computational cost, overlooking the impact of task placement on the network. This paper presents an online network-aware scheduler that is agnostic to the underlying scheduling policy and adjusts task allocation based on topological metrics and historical execution information. A parameter controls the weight of the network layer's intervention. Evaluated with the EASY, HEFT, CPOP, and PEFT heuristics on a Dragonfly topology, the proposed approach reduced the number of hops by up to 32%, inter-rack flows by 38%, inter-group flows by 51%, and estimated communication cost by 33%, with statistically significant improvements ($p < 0.05$).

1. Introdução

O crescimento dos sistemas de computação distribuída tem ampliado os desafios relacionados ao gerenciamento de recursos, à comunicação e ao escalonamento de aplicações complexas [TOP500.org 2025, Lu et al. 2022]. Aplicações podem ser estruturadas como *workflows*, geralmente representados por *Directed Acyclic Graphs* (DAGs), nos quais os vértices correspondem às tarefas e as arestas representam dependências e fluxos de dados. Plataformas como Apache Airflow, Apache Spark e Pegasus utilizam esse modelo para orquestrar execuções distribuídas [Deelman et al. 2019].

O escalonamento de *workflows* baseados em DAGs consiste em determinar onde e quando executar cada tarefa, respeitando suas dependências e a utilização de recursos. Heurísticas clássicas priorizam aspectos computacionais, tratando a comunicação de forma simplificada [Graham et al. 1979, Topcuoglu et al. 2002]. Assim, a integração entre escalonamento e estado da rede permanece um desafio em sistemas distribuídos [Liu and Buyya 2020, Saeedizade and Ashtiani 2025].

A comunicação pode impactar significativamente o desempenho quando tarefas dependentes são alocadas em diferentes servidores, *racks* ou grupos. O tráfego pode atravessar múltiplos enlaces e concentrar-se em determinados pontos da infraestrutura, aumentando o número de saltos, o custo de comunicação e a possibilidade de congestionamento [Noormohammadpour and Raghavendra 2018, Li et al. 2021, Ghabashneh et al. 2022, Canel et al. 2024].

Este trabalho propõe um escalonador de *workflows* consciente da rede, concebido como uma camada complementar e independente da política de escalonamento base. A abordagem preserva a ordenação das tarefas prontas, mas ajusta a seleção dos servidores com base em métricas topológicas, custos de comunicação e informações históricas de execuções anteriores. As principais contribuições são:

- um escalonador consciente da rede, complementar e independente da política base, que ajusta a seleção dos servidores sem alterar a ordenação das tarefas;
- a incorporação de informações históricas na seleção dos servidores candidatos, sempre reavaliadas pelo custo de comunicação atual;
- métricas orientadas à comunicação, incluindo fluxos entre servidores, *racks* e grupos, número de saltos e custo estimado;
- um parâmetro para controlar a intensidade da intervenção da camada de rede na decisão de alocação;
- uma avaliação por simulação em uma topologia *Dragonfly*, combinando a proposta com *EASY Backfilling*, HEFT, CPOP e PEFT.

A avaliação considera 4.000 *workflows* científicos, totalizando 21.360 tarefas, executados sobre uma topologia *Dragonfly* com 90 servidores e 9 *switches* [Juve et al. 2013, Kim et al. 2008]. Os resultados mostram reduções estatisticamente significativas de até 32,05% no número de saltos, 38,11% nos fluxos entre *racks*, 51,25% nos fluxos entre grupos e 32,95% no custo de comunicação ($p < 0,05$). Os ganhos diminuem à medida que a política base recebe maior peso, enquanto fluxos entre *racks* e grupos são reduzidos em todas as configurações.

Este artigo está organizado como segue: a Seção 2 apresenta os trabalhos relacionados; a Seção 3 descreve o ecossistema de simulação e as estratégias de escalonamento; a Seção 4 apresenta os resultados e discussões; e a Seção 5 apresenta as conclusões.

2. Trabalhos relacionados

O escalonamento de tarefas é um problema clássico em sistemas distribuídos. Entre as políticas propostas, destaca-se a *Extensible Argonne Scheduling System* (EASY), que segue a ordem de chegada e permite antecipar a execução de tarefas posteriores quando há recursos ociosos, desde que essa antecipação não atrase a primeira tarefa reservada na fila [Lifka 1995]. O *Predict Earliest Finish Time* (PEFT) é uma heurística baseada em lista que estima o impacto futuro de cada decisão: além do término imediato da tarefa corrente, considera uma estimativa otimista do custo restante do DAG, antecipando o efeito sobre as tarefas sucessoras [Arabnejad and Barbosa 2014]. O *Heterogeneous Earliest Finish Time* (HEFT) e o *Critical Path on a Processor* (CPOP), propostos por [Topcuoglu et al. 2002], baseiam-se em *ranks* de prioridade: o HEFT calcula um *rank* ascendente a partir de custos médios de computação e comunicação e aloca cada tarefa no recurso que oferece o menor tempo estimado de término, enquanto o CPOP combina os

ranks ascendente e descendente para identificar as tarefas do caminho crítico e mantê-las em um mesmo processador, reduzindo o tempo de conclusão desse caminho.

Heurísticas clássicas como o CPOP, variações de escalonamento por listas e o escalonamento estático de grafos foram amplamente estudadas em ambientes paralelos e heterogêneos [Graham et al. 1979, Kwok and Ahmad 1999]. Por priorizarem aspectos computacionais, *i.e.*, tempo de execução, ordem de chegada, caminho crítico, precedência e disponibilidade de recursos, podem produzir decisões desfavoráveis à comunicação, sobretudo quando tarefas dependentes são posicionadas em servidores distantes na topologia. Propostas que consideram localidade e posicionamento conjunto de computação e dados mostraram que incorporar informações de rede reduz tráfego e melhora o desempenho [Jalaparti et al. 2015], ainda que tendam a exigir planejamento prévio da carga ou modelos de infraestrutura mais complexos.

No contexto de *workflows*, [Shin et al. 2024] propõem escalonamento focado em nuvens com regra híbrida de prioridade e particionamento dinâmico de tarefas, [Koslovski et al. 2024] aplicam aprendizado por reforço ao escalonamento de DAGs, e [Saeedizade and Ashtiani 2025] sistematizam uma taxonomia de algoritmos para *workflows* científicos em nuvem. No contexto de *data centers*, [Vijayakumar et al. 2024] tratam da redução de latências de cauda em ambientes altamente utilizados e [Canel et al. 2024] analisam rajadas do tipo *incast*. Em conjunto, esses trabalhos indicam que a interação entre posicionamento de tarefas, comunicação e congestionamento permanece em aberto, pois esses trabalhos tratam o escalonamento de *workflows* e as questões de rede de *data center* de forma separada.

Cabe destacar que o levantamento bibliográfico não identificou escalonadores de *data center* baseados em *workflows* com consciência de rede, o que caracteriza a lacuna científica tratada por este trabalho. Assim, a proposta deste artigo preserva a lógica da política base e adiciona consciência de rede à alocação das tarefas, reduzindo custos lógicos de comunicação e potenciais pontos de congestionamento, sem alterar as dependências dos *workflows* nem a semântica de execução dos DAGs.

3. Escalonador consciente da rede

Neste trabalho, a campanha de simulação do escalonador proposto considera quatro políticas tradicionais de base: EASY, HEFT, CPOP e PEFT. O aspecto comum a essas quatro heurísticas é que, embora difiram na forma de ordenar as tarefas prontas e de estimar tempos de execução, todas respeitam as dependências do DAG sem considerar a topologia da rede nem o custo de comunicação ao escolher o servidor de execução. É nesse ponto que atua o escalonador consciente da rede proposto: ele preserva a ordenação definida pela política base e acrescenta, na etapa de seleção do servidor, as informações que as heurísticas tradicionais ignoram, *i.e.*, a topologia do *data center*, o posicionamento das tarefas predecessoras e o histórico de execuções anteriores.

Para cada nova tarefa, o escalonador identifica um conjunto de servidores candidatos e estima o custo de comunicação que seria gerado caso a tarefa fosse alocada em cada um deles. O conjunto de servidores candidatos reúne os servidores preferidos pela política base, os servidores previamente utilizados pela mesma tarefa em execuções anteriores e os servidores próximos aos predecessores da tarefa, com o intuito de minimizar comunicações. Em seguida, cada candidato recebe uma pontuação baseada na preferência

da política base e nas métricas de comunicação. O servidor escolhido é aquele que apresenta a menor pontuação final.

A pontuação utilizada é representada pela Equação 1, em que t representa a tarefa a ser escalonada, s representa um servidor candidato, $Score_{base}(s)$ representa a preferência do servidor segundo a política base, $Score_{net}(t, s)$ representa o custo estimado de comunicação e $\lambda \in [0, 1]$ controla o peso relativo dos componentes na decisão.

$$Score(t, s) = \lambda \times Score_{base}(s) + (1 - \lambda) \times Score_{net}(t, s) \quad (1)$$

Ambos os termos são normalizados no intervalo $[0, 1]$ entre os candidatos da mesma decisão, de modo que λ pondere grandezas comparáveis. $Score_{base}(s)$ corresponde à posição de s na ordenação de servidores preferida pela política base, dividida pelo número de candidatos; o servidor que a heurística tradicional escolheria recebe 0. Os extremos de λ delimitam o comportamento da camada: em $\lambda = 1$ a decisão é integralmente da política base, e a execução reproduz exatamente o *baseline*; em $\lambda = 0$ a decisão é integralmente conduzida pelo custo de comunicação, restrita ao conjunto de servidores livres no instante da alocação. O custo de rede $Score_{net}$ quantifica o impacto de comunicação de alocar uma tarefa em cada servidor candidato, a partir de métricas derivadas da topologia e do posicionamento das tarefas predecessoras. Para uma tarefa t e um servidor candidato s , ele corresponde à soma ponderada das métricas de comunicação normalizadas da Equação 2, em que $C_{server}(t, s)$, $C_{rack}(t, s)$ e $C_{group}(t, s)$ são as quantidades normalizadas de fluxos entre servidores, *racks* e grupos distintos, respectivamente, $C_{comm}(t, s)$ é o custo estimado de comunicação normalizado, e os pesos w_{server} , w_{rack} , w_{group} e w_{comm} controlam a importância relativa de cada métrica.

$$Score_{net}(t, s) = w_{server} \cdot C_{server}(t, s) + w_{rack} \cdot C_{rack}(t, s) + w_{group} \cdot C_{group}(t, s) + w_{comm} \cdot C_{comm}(t, s) \quad (2)$$

O histórico de execuções anteriores atua na composição do conjunto de servidores candidatos. Ao final de cada execução, o simulador registra, para cada tarefa, o servidor em que a tarefa foi posicionada. Em execuções subsequentes da mesma tarefa, esse servidor é incluído entre os candidatos, ao lado dos servidores preferidos pela política base e daqueles próximos aos predecessores, e é submetido ao mesmo cálculo de $Score_{net}$ que os demais. O posicionamento anterior é, portanto, revalidado à luz do estado corrente da topologia e da posição atual dos predecessores, e só é reutilizado quando continua sendo a melhor opção em termos de comunicação nas condições da execução em curso. Aproveitar o histórico dessa forma é coerente com nuvens privadas e *data centers* corporativos, nos quais *workflows* são executados de forma recorrente [Liu et al. 2011, Jalaparti et al. 2015]: posicionamentos já observados ampliam o espaço de busca da decisão, sem que o critério de escolha deixe de ser o custo de comunicação medido sobre a topologia.

O Algoritmo 1 resume a atuação do escalonador consciente da rede proposto. Trata-se de uma abordagem *online*, em que o posicionamento é decidido dinamicamente com base no estado corrente da topologia e no histórico observado. A ideia central é preservar as restrições de precedência dos DAGs e modificar apenas a escolha do servidor de

Algorithm 1 Simulação do escalonamento de DAGs com consciência da rede, combinada com uma política base

```
1: Inicializar workflows, servidores, topologia e histórico
2: Definir  $P_{base} \in \{\text{EASY}, \text{HEFT}, \text{CPOP}, \text{PEFT}\}$ 
3: while existirem tarefas não concluídas do
4:   Liberar DAGs disponíveis no tempo atual
5:   Atualizar tarefas finalizadas e servidores livres
6:   Identificar tarefas prontas  $T_{ready}$ 
7:   Ordenar  $T_{ready}$  conforme  $P_{base}$ 
8:   for all  $t \in T_{ready}$  do
9:     Obter servidores livres  $S_{free}$ 
10:    if  $S_{free} = \emptyset$  then
11:      Interromper escalonamento da iteração
12:    end if
13:    Selecionar candidatos  $S_{cand} \subseteq S_{free}$  (preferidos por  $P_{base}$ , vizinhos dos predecessores e posicionamento histórico de  $t$ )
14:    Calcular métricas de rede para cada  $s \in S_{cand}$ 
15:    Combinar preferência da política base e custo de rede
16:    Selecionar  $s^* = \arg \min_{s \in S_{cand}} \text{Score}(t, s)$ 
17:    Escalonar  $t$  em  $s^*$  e registrar métricas
18:  end for
19:  Avançar o tempo da simulação
20: end while
```

destino segundo métricas de custo de comunicação: duas tarefas podem ser selecionadas na mesma ordem definida pela política base, mas ter seu posicionamento ajustado para reduzir fluxos entre *racks*, grupos ou servidores distantes na topologia. A proposta não contempla preempção, *i.e.*, uma vez iniciada em um servidor, a tarefa executa até a conclusão, sem interrupções motivadas pela camada consciente da rede. Da mesma forma, inanição não é tratada como preocupação específica, pois a ordem de seleção das tarefas prontas permanece integralmente definida pela política base, que já resolve esse aspecto.

4. Análise experimental e resultados

A análise experimental baseou-se no ambiente de simulação desenvolvido em Python¹ para avaliar políticas de escalonamento de *workflows* em uma topologia de *data center*. O objetivo deste ambiente é permitir a análise de diferentes combinações de heurísticas tradicionais e métricas representativas da perspectiva da rede. A carga aplicada a este ambiente é constituída por DAGs derivados de perfis de *workflows* científicos caracterizados na literatura e utilizados no contexto do projeto Pegasus [Juve et al. 2013, Deelman et al. 2019], resultando em 4.000 DAGs e 21.360 tarefas, que saturam a infraestrutura e mantêm a fila de tarefas prontas permanentemente populada.

Cada *workflow* é descrito como um conjunto de tarefas com tempo de submissão, tempo estimado de execução, consumo de recurso e dependências explícitas entre tarefas. Por sua vez, a infraestrutura de execução é representada por uma topologia *Dra-*

¹Disponível em <https://github.com/victortalvarenga-arch/mestrado> (Repositório público na versão final do artigo).

gonfly [Kim et al. 2008]. Essa topologia foi proposta para interconexões de larga escala, explorando roteadores de alta radix, baixa distância média entre nós e organização hierárquica em grupos. Servidores de computação são organizados em *racks*, conectados a roteadores locais, que por sua vez são agrupados em domínios de comunicação, permitindo distinguir enlaces locais, enlaces entre *racks* de um mesmo grupo e enlaces entre grupos distintos. Essa estrutura é relevante para a análise do escalonador proposto, pois permite estimar diferentes níveis de custo de comunicação entre tarefas dependentes.

4.1. Plano de testes

Foram avaliadas quatro políticas base de escalonamento; sobre cada política base, foram executadas quatro configurações de pesos das métricas de rede, além da execução *baseline* sem consciência de rede (Tabela 1). Cada uma dessas configurações foi replicada para cinco valores de λ (0, 0,25, 0,5, 0,75 e 1), totalizando 80 execuções conscientes da rede (4 heurísticas $\times$ 4 configurações de pesos $\times$ 5 valores de λ) comparadas às 4 execuções *baseline* correspondentes (Tabela 2). A varredura de λ percorre todo o intervalo de intervenção da camada proposta e inclui deliberadamente o extremo $\lambda = 1$, que serve como verificação de sanidade: nesse ponto, a decisão é integralmente da política base e a execução deve reproduzir o *baseline* tarefa a tarefa.

Tabela 1. Configurações de pesos avaliadas durante as simulações.

Configuração	Descrição da configuração de pesos
<i>baseline</i>	Execução da política base sem consciência de rede.
<i>balanced</i>	Pesos balanceados entre as métricas de rede.
<i>rack strict</i>	Maior penalização para fluxos entre <i>racks</i> distintos.
<i>group strict</i>	Maior penalização para fluxos entre grupos distintos.
<i>comm cost strict</i>	Maior penalização para o custo estimado de comunicação.

Quanto às métricas, foram analisados a quantidade total de fluxos de comunicação entre tarefas dependentes, o número total de saltos percorridos na topologia, os fluxos entre servidores, *racks* e grupos distintos, o custo estimado de comunicação, a média de saltos por fluxo e a utilização média dos recursos computacionais. Na leitura das métricas de comunicação, valores menores indicam melhor desempenho, pois representam menor tráfego atravessando diferentes níveis da topologia. Reduções na quantidade de fluxos e no número de saltos evidenciam que o escalonador teve êxito em aproximar tarefas dependentes, diminuindo o custo lógico de comunicação.

4.2. Discussão quantitativa

A variável λ pondera a interferência do escalonador consciente de rede na seleção do servidor: quanto menor λ , maior o peso da métrica de rede na decisão. As Figuras 1 a 3 detalham os cenários de maior atuação ($0 \leq \lambda \leq 0,50$), e a Tabela 2 consolida as 64 combinações de heurística, configuração e λ , expressas como ganho percentual médio por tarefa em relação ao *baseline* ($\lambda = 1$). Sobre o conjunto completo, o ganho variou de $-1,71\%$ a $32,05\%$ em número de saltos (*hops*), de $-5,82\%$ a $22,84\%$ em fluxos entre servidores (*cross-srv*), de $2,47\%$ a $38,11\%$ em fluxos entre *racks* (*cross-rack*), de $2,45\%$ a $51,51\%$ em fluxos entre grupos (*cross-grp*) e de $-1,80\%$ a $33,20\%$ no custo estimado de comunicação. Os extremos superiores concentram-se na EASY sob $\lambda \leq 0,25$ e os inferiores no HEFT sob $\lambda = 0,75$; na EASY, os ganhos em número de saltos sob $\lambda \leq 0,25$

mantiveram-se iguais ou superiores a 27,68%. Perdas em *cross-srv* ocorrem apenas para $\lambda \geq 0,50$ (18 das 32 combinações), ao passo que em fluxos entre *racks* (*cross-rack*) e em fluxos entre grupos (*cross-grp*) permanecem positivos em toda a faixa avaliada.

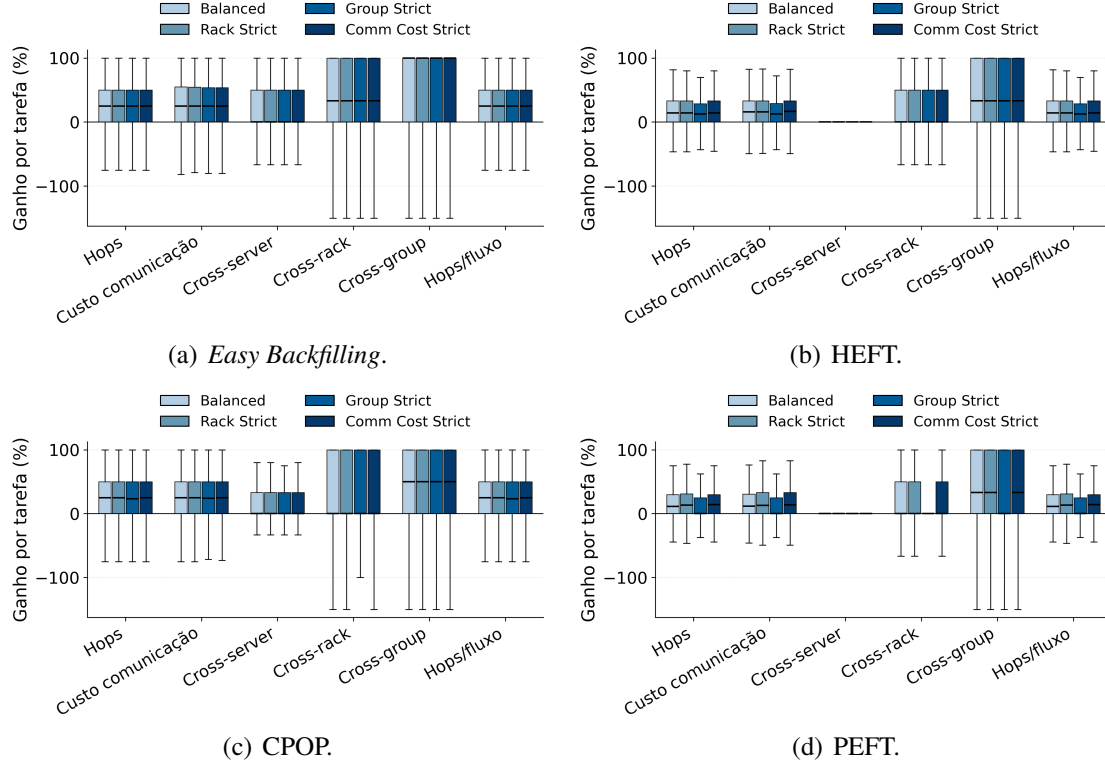
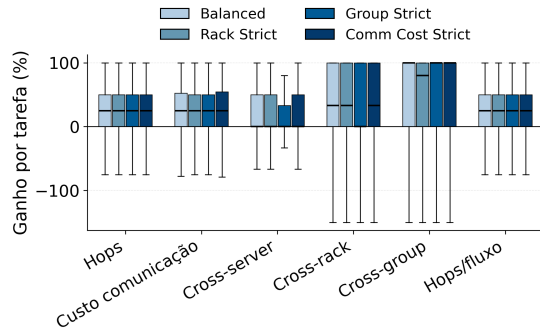


Figura 1. Ganho por tarefa nas métricas de comunicação, com $\lambda = 0$.

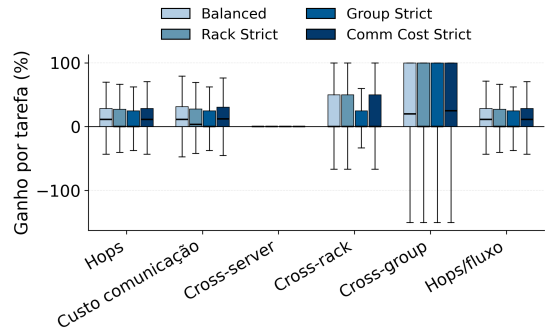
No ponto de operação $\lambda = 0$, todos os ganhos são positivos: número de saltos (*hops*) de 10,93% a 32,05%; *cross-srv* de 0,88% a 22,84%; fluxos entre *racks* (*cross-rack*) de 9,49% a 38,11%; fluxos entre grupos (*cross-grp*) de 19,65% a 51,25%; e custo de comunicação de 11,60% a 32,95%. O maior efeito nos fluxos entre *racks* e entre grupos ocorre na EASY sob *rack strict*, e o menor, no PEFT sob *group strict*. As 80 comparações desse ponto de operação foram estatisticamente significativas. A utilização média do *cluster* permaneceu idêntica à do *baseline* nas 64 execuções conscientes da rede, para todos os valores de λ : 2.431 unidades de tempo e utilização de 98,53% para a EASY; e 2.416 unidades de tempo e utilização de 99,14% para HEFT, CPOP e PEFT. O resultado decorre do desenho da proposta, que reordena apenas a escolha entre servidores já ociosos, sem alterar a ordem das tarefas prontas nem o instante de despacho.

4.3. Sensibilidade ao parâmetro de controle λ

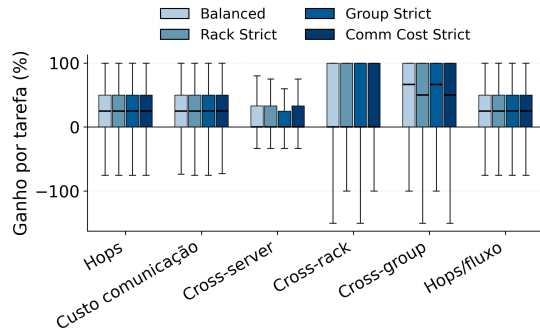
As Figuras 4 a 6 sintetizam a varredura em mapas de calor: os painéis correspondem às configurações de pesos; as linhas, às heurísticas-base; as colunas, a λ ; e as células, ao ganho médio percentual por tarefa, em verde (redução) ou vermelho (aumento). O padrão dominante é o decaimento ordenado dos ganhos conforme λ cresce. Na EASY sob *balanced*, a redução de saltos passa de 32,05% em $\lambda = 0$ para 30,22%, 20,84% e 3,35% em $\lambda = 0,25, 0,5$ e $0,75$. O decaimento não é linear: sobre as 80 comparações, a queda média é de 0,99 ponto percentual entre $\lambda = 0$ e $0,25$, contra 6,49 entre $0,25$ e $0,5$ e 13,09 entre $0,5$ e $0,75$. A participação minoritária da política base mantém a decisão



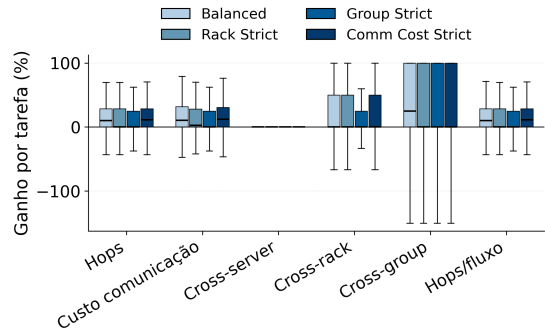
(a) *Easy Backfilling*.



(b) HEFT.

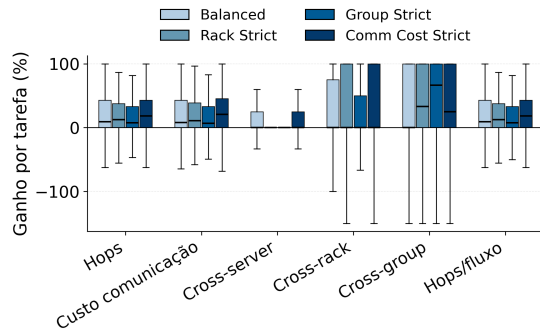


(c) CPOP.

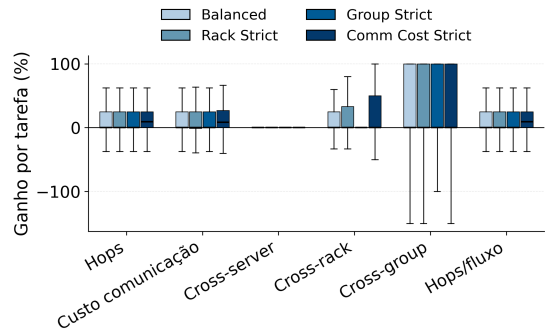


(d) PEFT.

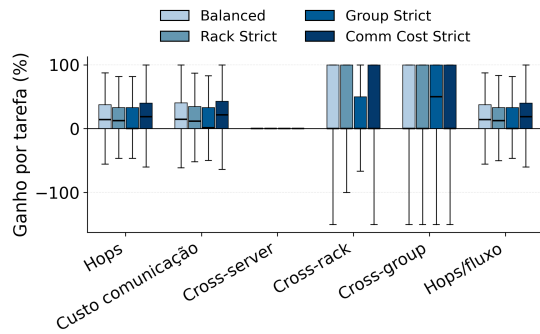
Figura 2. Ganho por tarefa nas métricas de comunicação, com $\lambda = 0,25$.



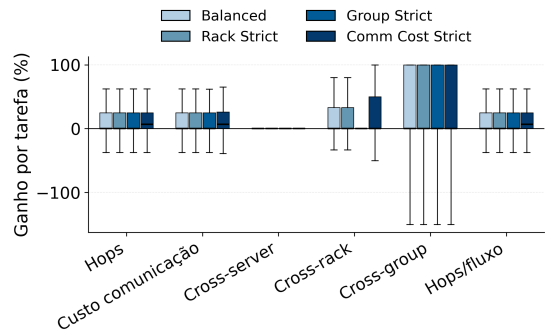
(a) *Easy Backfilling*.



(b) HEFT.



(c) CPOP.



(d) PEFT.

Figura 3. Ganho por tarefa nas métricas de comunicação, com $\lambda = 0,5$.

predominantemente determinada pela camada de rede, e os maiores ganhos concentram-se em $\lambda \leq 0,25$. O CPOP é a exceção ao decaimento monotônico: sob *balanced* e *comm cost strict*, seus maiores ganhos ocorrem em $\lambda = 0,25$, com reduções de 26,73% e 26,94% nos saltos, contra 24,62% e 24,72% em $\lambda = 0$. Como o CPOP mantém as tarefas do caminho crítico em um mesmo processador, preservar parte dessa preferência restringe a busca a posicionamentos que também reduzem a comunicação; descartar essa preferência dispersa essas tarefas.

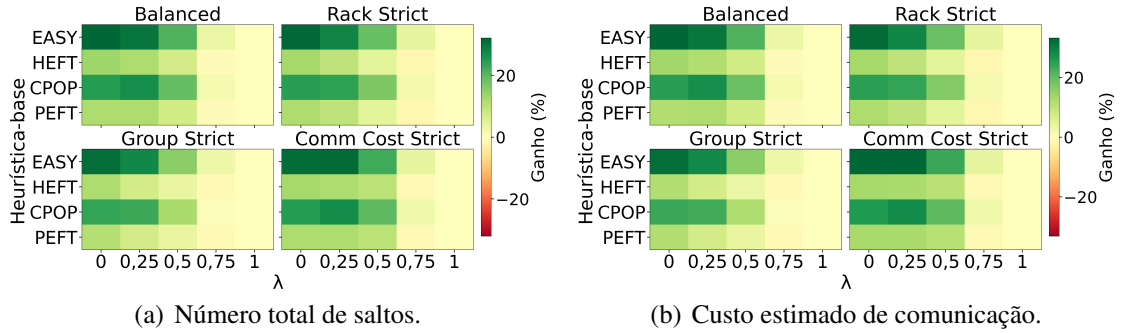


Figura 4. Ganho médio por tarefa em função de λ , nas métricas agregadas.

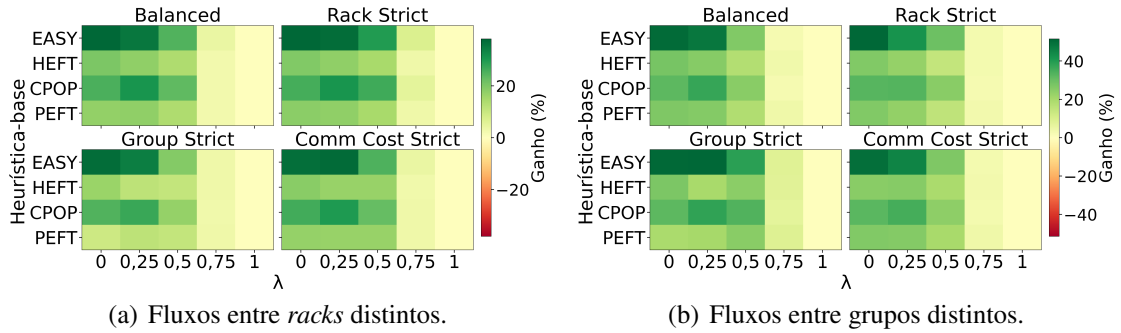


Figura 5. Ganho médio por tarefa em função de λ : fluxos entre *racks* e grupos.

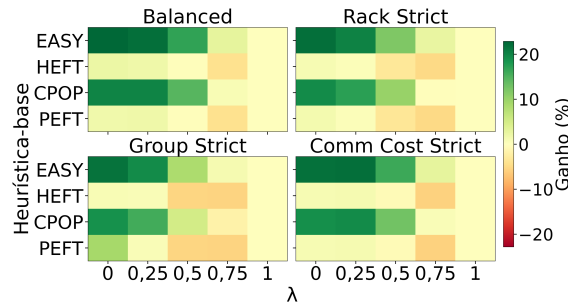


Figura 6. Ganho médio por tarefa em função de λ , nos fluxos entre servidores.

Dois pontos merecem registro. Primeiro, os fluxos entre *racks* e entre grupos foram reduzidos nas 64 combinações com $\lambda < 1$: mesmo em $\lambda = 0,75$, quando a política base já domina a decisão, as reduções mínimas são de 2,47% e 2,45%. Esses são os deslocamentos de maior custo na topologia *Dragonfly*, e o efeito mantém-se em toda a faixa de λ . Segundo, o efeito de compensação nos fluxos entre servidores associa-se a valores intermediários e altos de λ , e não a uma característica intrínseca do HEFT e do PEFT: em $\lambda \leq 0,25$ nenhuma das 32 combinações piora nessa métrica, e sob *balanced* ambos ganham 2,16% e 1,78% em $\lambda = 0$. O efeito surge em $\lambda = 0,5$, restrito a HEFT e PEFT, com perdas de até 5,43% e 5,40% sob *group strict*, e intensifica-se em $\lambda = 0,75$,

alcançando 5,82% no HEFT sob *comm cost strict* e afetando também o CPOP. Com pouco peso, a camada de rede apenas aproxima tarefas dependentes no mesmo *rack* ou grupo, convertendo fluxos entre *racks* e entre grupos em fluxos entre servidores; com peso total, reorganiza o posicionamento. Em $\lambda = 0,75$, HEFT e PEFT apresentam piora nos saltos e no custo de comunicação, entre 0,67% e 1,80%, ainda que sigam reduzindo fluxos entre *racks* e entre grupos. É o único regime em que a proposta degrada uma métrica agregada, situando λ como parâmetro de configuração da política e não como detalhe de implementação. Nesse ponto, a mediana e ambos os quartis de CPOP e PEFT são nulos em todas as métricas: 55,8% e 63,4% das tarefas mantêm o posicionamento do *baseline* e as alterações restantes concentram-se nas caudas, não exibidas; os ganhos médios da Tabela 2 provêm dessa fração.

4.4. Análise de significância estatística

As tarefas das execuções *baseline* e consciente da rede foram pareadas por identificador e as diferenças por tarefa submetidas ao teste não paramétrico de Wilcoxon pareado, ao nível de 5% ($p < 0,05$). A Tabela 2 reporta, para cada λ , heurística e configuração de pesos, o ganho médio percentual por tarefa sobre o *baseline*: valores positivos indicam redução (melhora) da métrica e negativos, aumento (piora); a coluna Custo corresponde ao custo estimado de comunicação. Os valores de p não são apresentados; os resultados que não atingiram o nível de significância estão destacados em vermelho. Das 320 comparações com $\lambda < 1$, 309 foram estatisticamente significativas. As onze exceções restringem-se aos fluxos entre servidores (sete casos) e ao custo de comunicação (quatro), nenhuma na EASY, e correspondem a variações inferiores a 1,3% em valor absoluto, o que caracteriza ausência de efeito detectável, e não perda de desempenho. Os ganhos mantêm-se estáveis entre heurísticas, configurações de pesos e valores de λ , o que caracteriza efeito sistemático, e não variações pontuais. O aumento nos fluxos entre servidores observado em parte das configurações corresponde ao efeito de compensação discutido na Seção 4.3, restrito a $\lambda \geq 0,5$ e concentrado no HEFT e no PEFT.

4.5. Discussão comparativa

Embora as heurísticas se beneficiem da política consciente da rede, cada uma possui pontos fortes e limitações inerentes à sua lógica de escalonamento (Tabela 3). A possibilidade de priorizar métricas específicas, por meio das configurações de pesos, e de calibrar a intensidade da intervenção, por meio de λ , confere flexibilidade à abordagem para atender diferentes objetivos operacionais. A ordenação relativa das heurísticas mantém-se estável ao longo de toda a varredura de λ : a EASY obtém sistematicamente os maiores ganhos, seguida pelo CPOP, com HEFT e PEFT apresentando as menores margens de melhoria. De forma geral, a proposta aproxima o processo de escalonamento das características reais da infraestrutura de rede, permitindo que decisões considerem não apenas disponibilidade computacional e precedência entre tarefas, mas também o impacto topológico das comunicações geradas durante a execução dos *workflows*.

5. Conclusão

Este trabalho apresentou uma política de escalonamento de *workflows* consciente da rede. A proposta atua de forma complementar sobre políticas tradicionais de escalonamento. As simulações avaliaram quatro heurísticas base (EASY, HEFT, CPOP e PEFT), combinadas com quatro configurações de pesos das métricas de comunicação e cinco valores

Tabela 2. Ganho médio por tarefa (*baseline* vs. consciente da rede).

Heurística	Configuração	λ	Hops	Cross-srv	Cross-rack	Cross-grp	Custo
EASY	Balanced	0	32,05%	22,84%	37,96%	50,61%	32,95%
		0,25	30,22%	22,06%	35,60%	48,00%	31,07%
		0,5	20,84%	16,91%	25,00%	26,35%	21,66%
		0,75	3,35%	3,01%	3,98%	2,45%	3,35%
	Rack Strict	0	31,73%	22,11%	38,11%	51,25%	32,64%
		0,25	28,19%	20,21%	37,15%	41,94%	29,04%
		0,5	18,88%	11,90%	29,75%	29,40%	19,23%
		0,75	3,95%	2,58%	6,86%	2,97%	3,92%
	Group Strict	0	31,00%	21,61%	36,99%	51,09%	31,94%
		0,25	27,68%	19,28%	34,34%	51,51%	28,44%
		0,5	15,50%	8,56%	19,07%	38,72%	15,87%
		0,75	2,66%	1,18%	3,21%	8,10%	2,63%
	Comm Cost Strict	0	31,15%	21,80%	36,81%	49,83%	32,95%
		0,25	31,52%	22,00%	37,22%	45,25%	33,20%
		0,5	22,42%	16,09%	25,07%	26,91%	23,66%
		0,75	3,99%	2,98%	3,99%	2,92%	4,27%
HEFT	Balanced	0	13,43%	2,16%	19,95%	28,04%	13,57%
		0,25	11,62%	1,89%	17,73%	25,49%	11,92%
		0,5	7,24%	-0,28%	13,86%	17,86%	7,55%
		0,75	-0,93%	-4,15%	2,47%	3,66%	-0,85%
	Rack Strict	0	12,72%	1,05%	19,61%	26,51%	13,00%
		0,25	9,60%	0,37%	17,81%	23,00%	9,92%
		0,5	4,50%	-3,26%	14,61%	14,63%	4,57%
		0,75	-1,71%	-5,14%	3,10%	2,84%	-1,80%
	Group Strict	0	11,54%	0,88%	16,63%	27,06%	11,87%
		0,25	7,23%	0,56%	11,72%	19,84%	7,37%
		0,5	3,68%	-5,43%	10,89%	24,69%	3,58%
		0,75	-1,17%	-5,61%	3,10%	7,95%	-1,18%
	Comm Cost Strict	0	12,55%	1,05%	18,74%	25,13%	13,22%
		0,25	11,55%	1,13%	16,88%	25,36%	12,45%
		0,5	10,21%	-0,71%	16,85%	19,36%	10,84%
		0,75	-1,17%	-5,82%	2,94%	3,56%	-1,22%
CPOP	Balanced	0	24,62%	20,02%	25,48%	31,84%	25,34%
		0,25	26,73%	20,03%	30,82%	37,31%	27,58%
		0,5	19,04%	14,53%	23,30%	25,34%	19,32%
		0,75	1,74%	0,83%	2,65%	2,60%	1,58%
	Rack Strict	0	24,50%	19,41%	25,94%	32,75%	25,12%
		0,25	23,94%	17,18%	30,80%	33,20%	24,22%
		0,5	16,58%	10,00%	26,55%	25,35%	16,81%
		0,75	1,86%	-0,30%	5,31%	2,56%	1,75%
	Group Strict	0	23,36%	18,89%	24,89%	31,81%	23,75%
		0,25	22,56%	15,83%	27,36%	37,83%	23,02%
		0,5	12,21%	5,04%	17,14%	33,52%	12,10%
		0,75	0,63%	-1,82%	2,87%	7,24%	0,62%
	Comm Cost Strict	0	24,72%	18,97%	26,46%	32,23%	26,13%
		0,25	26,94%	19,33%	29,97%	35,17%	28,25%
		0,5	19,81%	12,71%	22,77%	24,24%	20,61%
		0,75	2,43%	0,58%	2,99%	2,87%	2,58%
PEFT	Balanced	0	11,58%	1,78%	17,31%	26,96%	11,88%
		0,25	11,64%	1,89%	17,76%	26,00%	12,06%
		0,5	7,29%	-0,27%	13,95%	18,42%	7,75%
		0,75	-0,88%	-4,11%	2,52%	4,30%	-0,67%
	Rack Strict	0	11,67%	1,29%	18,21%	26,22%	12,18%
		0,25	9,62%	0,37%	17,84%	23,53%	10,07%
		0,5	4,42%	-3,47%	14,67%	15,30%	4,62%
		0,75	-1,69%	-5,14%	3,15%	3,50%	-1,63%
	Group Strict	0	10,93%	9,01%	9,49%	19,65%	11,60%
		0,25	7,25%	0,56%	11,76%	20,39%	7,52%
		0,5	3,73%	-5,40%	11,01%	25,21%	3,79%
		0,75	-1,22%	-5,70%	3,11%	8,54%	-1,09%
	Comm Cost Strict	0	11,52%	1,03%	17,11%	26,23%	12,74%
		0,25	11,57%	1,13%	16,91%	25,86%	12,59%
		0,5	10,22%	-0,71%	16,88%	19,91%	10,98%
		0,75	-1,11%	-5,78%	2,99%	4,22%	-1,04%

do parâmetro λ , que pondera a política base e a camada consciente da rede, totalizando 80 execuções. Os resultados obtidos indicam que a incorporação de consciência de rede no processo de escalonamento melhora consistentemente o perfil de comunicação dos *workflows*, independentemente da heurística base utilizada. Em $\lambda = 0$, ponto em que a decisão de posicionamento é integralmente delegada à camada de rede, observou-se redução simultânea em todas as métricas avaliadas, com ganhos médios por tarefa de até

Tabela 3. Resumo qualitativo das heurísticas combinadas com a política consciente da rede.

Heurística	Pontos Fortes	Limitações / Observações
EASY	Simplicidade de implementação; maiores ganhos relativos entre as heurísticas, com melhorias consistentes em todas as métricas e em todos os valores de λ .	Política base gera mais comunicação, o que amplia a margem de melhoria da consciência da rede.
HEFT	Reduz significativamente saltos e custo de comunicação com desempenho superior em <i>workflows</i> heterogêneos; em $\lambda \leq 0,25$ melhora também os fluxos entre servidores.	Algoritmo complexo; a partir de $\lambda \geq 0,5$ passa a aumentar fluxos locais entre servidores (efeito de compensação) e, em $\lambda = 0,75$, degrada levemente saltos e custo.
CPOP	Mantém tarefas críticas próximas com redução significativa em fluxos entre <i>racks</i> e ganhos consistentes sob alta concorrência; único caso em que o melhor resultado ocorre em $\lambda = 0,25$, e não em $\lambda = 0$.	Sensível à parametrização de candidatos; sobrecarga maior para <i>workflows</i> com várias tarefas.
PEFT	Alocação base já eficiente em comunicação.	Menor ganho relativo entre as heurísticas, pois a política base já explora localidade; também sujeita ao efeito de compensação em fluxos entre servidores para $\lambda \geq 0,5$.

32,05% no número de saltos, 22,84% nos fluxos entre servidores, 38,11% nos fluxos entre *racks*, 51,25% nos fluxos entre grupos e 32,95% no custo estimado de comunicação. A varredura de λ mostrou que os fluxos entre *racks* e entre grupos, associados aos deslocamentos mais custosos na topologia, são reduzidos em todas as combinações avaliadas, e que o efeito de compensação nos fluxos entre servidores é um fenômeno de regimes intermediários, e não uma limitação intrínseca das heurísticas base. Em nenhuma das execuções houve alteração da utilização do *cluster*, o que confirma que os ganhos de comunicação foram obtidos sem custo em desempenho global. Dessa forma, a política proposta aproxima o processo de orquestração das características reais da infraestrutura de comunicação, mantendo compatibilidade com políticas tradicionais e permitindo sua extensão para diferentes estratégias de decisão.

Cabe registrar que a avaliação restringe-se a um cenário de carga e a uma instância de topologia *Dragonfly*, e que as métricas de comunicação são estimadas a partir do modelo topológico, sem simulação de contenção em enlaces. Como trabalhos futuros, pretende-se incorporar medições efetivas de latência, *buffers*, largura de banda e ocupação de enlaces, avaliar a proposta sobre outras topologias e cargas, bem como explorar mecanismos automáticos para ajuste dinâmico de λ e dos pesos das métricas de rede, dado que o ponto de operação ótimo mostrou-se dependente da heurística base.

Agradecimentos

O trabalho foi desenvolvido no LabP2D, com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (PROAP/AUXPE) (CAPES) (Código de Financiamento 001), Fundação de Amparo à Pesquisa e Inovação do Estado de Santa Catarina (FAPESC) e Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq).

Referências

Arabnejad, H. and Barbosa, J. G. (2014). List scheduling algorithm for heterogeneous systems by an optimistic cost table. *IEEE Transactions on Parallel and Distributed Systems*, 25(3):682–694.

- Canel, C., Madhavan, B., Sundaresan, S., Spring, N., Kannan, P., Zhang, Y., Lin, K., and Seshan, S. (2024). Understanding incast bursts in modern datacenters. In *Proceedings of the 2024 ACM Internet Measurement Conference*, pages 674–680. ACM.
- Deelman, E., Vahi, K., Rynge, M., Mayani, R., da Silva, R. F., Papadimitriou, G., and Livny, M. (2019). The evolution of the pegasus workflow management software. *Computing in Science & Engineering*, 21(4):22–36.
- Ghabashneh, E., Zhao, Y., Lumezanu, C., Spring, N., Sundaresan, S., and Rao, S. (2022). A microscopic view of bursts, buffer contention, and loss in data centers. In *Proceedings of the 22nd ACM Internet Measurement Conference*, pages 567–580. ACM.
- Graham, R. L., Lawler, E. L., Lenstra, J. K., and Rinnooy Kan, A. H. G. (1979). Optimization and approximation in deterministic sequencing and scheduling: A survey. *Annals of Discrete Mathematics*, 5:287–326.
- Jalaparti, V., Bodik, P., Menache, I., Rao, S., Makarychev, K., and Caesar, M. (2015). Network-aware scheduling for data-parallel jobs: Plan when you can. In *Proceedings of the 2015 ACM Conference on Special Interest Group on Data Communication*, pages 407–420. ACM.
- Juve, G., Chervenak, A., Deelman, E., Bharathi, S., Mehta, G., and Vahi, K. (2013). Characterizing and profiling scientific workflows. *Future Generation Computer Systems*, 29(3):682–692.
- Kim, J., Dally, W. J., Scott, S., and Abts, D. (2008). Technology-driven, highly-scalable dragonfly topology. In *Proceedings of the 35th Annual International Symposium on Computer Architecture*, pages 77–88. IEEE Computer Society.
- Koslovski, G. P., Pereira, K., and Albuquerque, P. R. (2024). Dag-based workflows scheduling using actor-critic deep reinforcement learning. *Future Generation Computer Systems*, 150:354–363.
- Kwok, Y.-K. and Ahmad, I. (1999). Static scheduling algorithms for allocating directed task graphs to multiprocessors. *ACM Computing Surveys*, 31(4):406–471.
- Li, W., Liu, J., Wang, S., Zhang, T., Zou, S., Hu, J., Jiang, W., and Huang, J. (2021). Survey on traffic management in data center network: From link layer to application layer. *IEEE Access*, 9:38427–38456.
- Lifka, D. A. (1995). The anl/ibm sp scheduling system. In *Proceedings of the Workshop on Job Scheduling Strategies for Parallel Processing*, pages 295–303, Santa Barbara, CA, USA. Springer.
- Liu, F., Tong, J., Mao, J., Bohn, R., Messina, J., Badger, L., and Leaf, D. (2011). Nist cloud computing reference architecture. Special Publication 500-292, National Institute of Standards and Technology, Gaithersburg, MD, USA.
- Liu, X. and Buyya, R. (2020). Resource management and scheduling in distributed stream processing systems: A taxonomy, review, and future directions. *ACM Computing Surveys*, 53(3):1–40.
- Lu, P.-J., Lai, M.-C., and Chang, J.-S. (2022). A survey of high-performance interconnection networks in high-performance computer systems. *Electronics*, 11(9):1369.

- Noormohammadpour, M. and Raghavendra, C. S. (2018). Datacenter traffic control: Understanding techniques and tradeoffs. *IEEE Communications Surveys & Tutorials*, 20(2):1492–1525.
- Saeedizade, E. and Ashtiani, M. (2025). Scientific workflow scheduling algorithms in cloud environments: A comprehensive taxonomy, survey, and future directions. *Journal of Scheduling*, 28(1):1–63.
- Shin, J., Arroyo, D., Tantawi, A. N., Wang, C., Youssef, A., and Nagi, R. (2024). Cloud-native workflow scheduling using a hybrid priority rule, dynamic resource allocation, and dynamic task partition. In *Proceedings of the 2024 ACM Symposium on Cloud Computing*, pages 830–846. ACM.
- TOP500.org (2025). El capitan retains #1 as jupiter becomes europe’s first exascale system in the 66th top500 list. <https://www.top500.org/>. Accessed: 2026-06-14.
- Topcuoglu, H., Hariri, S., and Wu, M.-Y. (2002). Performance-effective and low-complexity task scheduling for heterogeneous computing. *IEEE Transactions on Parallel and Distributed Systems*, 13(3):260–274.
- Vijayakumar, S., Madhavapeddy, A., and Kalyvianaki, E. (2024). Scheduling for reduced tail task latencies in highly utilized datacenters. In *Proceedings of the 2024 ACM Symposium on Cloud Computing*, pages 302–321. ACM.