Uma Análise Abrangente da Seleção de Algoritmos: Dos Fundamentos Teóricos aos Portfólios Interpretáveis e 'Anytime'

Secção 1: A Base Formal do Problema de Seleção de Algoritmos

Esta secção estabelece a justificação fundamental e o enquadramento formal para a seleção de algoritmos. Argumenta-se que a necessidade de seleção não é uma questão de conveniência, mas uma necessidade teórica, e introduz-se a linguagem formal utilizada para discutir o problema.

1.1 O Imperativo Teórico: 'No Free Lunch' em Otimização

A busca por um único algoritmo de otimização universalmente superior é uma ambição antiga na ciência da computação. No entanto, os teoremas "No Free Lunch" (NFL), ou "Não Há Almoço Grátis", fornecem uma refutação matemática robusta a esta noção.[1] O conceito central dos teoremas NFL afirma que, para qualquer algoritmo de otimização, o desempenho elevado em uma certa classe de problemas é inevitavelmente compensado por um desempenho inferior em outra classe. Quando o desempenho é calculado como uma média sobre

todos os problemas possíveis, todos os algoritmos de otimização apresentam um desempenho médio idêntico.[1] Esta constatação estabelece o argumento fundamental contra a existência de um algoritmo "bala de prata" e serve como a principal justificação teórica para o campo da seleção de algoritmos.

No entanto, uma análise crítica revela que a aplicabilidade direta dos teoremas NFL ao mundo real é limitada por uma suposição central, mas muitas vezes irrealista: a de que os problemas

são extraídos de uma distribuição de probabilidade uniforme sobre o espaço de *todos os problemas possíveis*.[4] Na prática, os problemas que encontramos em domínios específicos, como o particionamento de grafos para redes sociais ou o planeamento logístico, não representam uma amostra aleatória e uniforme de todos os problemas matemáticos concebíveis. Pelo contrário, constituem um subconjunto pequeno e altamente estruturado. Por exemplo, os grafos que representam redes sociais exibem propriedades específicas, como distribuições de grau do tipo lei de potência e alta clusterização, que os distinguem de um grafo aleatório genérico.[6]

Esta violação da suposição de uniformidade é precisamente o que torna a seleção de algoritmos não só possível, mas necessária. Se o domínio do problema é estruturado, então é viável que um algoritmo explore essa estrutura para obter um desempenho superior. Os teoremas NFL não proíbem um "almoço grátis" num subconjunto específico e relevante de problemas; pelo contrário, eles implicam que a busca por tal "almoço" deve ser direcionada e informada pela estrutura do problema em questão.[4] A utilidade prática dos teoremas NFL, portanto, não é desencorajar a busca por melhores algoritmos, mas sim forçar uma mudança de foco: da criação de algoritmos universalmente superiores para a criação de

seletores superiores. O desafio central da algoritmia moderna não é apenas o design de algoritmos, mas o design de mecanismos de seleção que possam mapear as características de um problema para o algoritmo mais adequado para o resolver.

1.2 Uma Estrutura Formal: O Modelo de Rice para a Seleção de Algoritmos

A primeira formalização rigorosa do Problema de Seleção de Algoritmos (ASP, do inglês *Algorithm Selection Problem*) foi proposta por John R. Rice em 1976, fornecendo uma estrutura conceptual que continua a ser a base do campo.[7] O modelo de Rice decompõe o problema em quatro componentes ou espaços interligados, que, juntos, definem o ecossistema no qual a seleção ocorre.

O modelo de Rice articula o ASP da seguinte forma: para uma dada instância de problema x, extraímos um vetor de características f(x). O objetivo é encontrar um mapeamento de seleção S(f(x)) que escolhe um algoritmo α do espaço de algoritmos, de tal forma que o desempenho y(α(x)) seja otimizado.[9] Este enquadramento transforma a seleção de algoritmos de uma arte heurística para um problema bem definido que pode ser abordado com as ferramentas da matemática e da aprendizagem de máquina. A Tabela 1 detalha cada um dos espaços de Rice, utilizando o problema de particionamento de grafos como um exemplo concreto e recorrente para ilustrar os conceitos abstratos.

Tabela 1: A Estrutura de Seleção de Algoritmos de Rice com Exemplos de Particionamento de Grafos

Espaço	Definição	Exemplo em Particionamento de Grafos
Espaço do Problema (P)	O conjunto de todas as instâncias possíveis de um problema computacional.	O conjunto de todos os grafos que podem ser particionados, desde redes sociais e biológicas até redes rodoviárias e malhas de elementos finitos.[16]
Espaço de Características (F)	O conjunto de características numéricas mensuráveis que descrevem uma instância do problema.	Características do grafo como número de vértices ($
Espaço de Algoritmos (A)	O conjunto de algoritmos candidatos disponíveis para resolver o problema.	Um portfólio de particionadores de grafos, como METIS, heurísticas gulosas, Recozimento Simulado (Simulated Annealing), Algoritmos Genéticos, etc..[18]
Espaço de Desempenho (Y)	O mapeamento do desempenho de cada algoritmo numa dada instância, medido por uma métrica específica.	Métricas de qualidade da partição, como o tamanho do corte (número de arestas entre partições) ou a modularidade, e métricas de custo, como o tempo de CPU ou o uso de memória.[9]

A operacionalização do modelo de Rice depende criticamente da capacidade de definir e calcular características informativas no Espaço de Características. Estas características servem como a ponte entre a instância do problema e a decisão de seleção. Para o particionamento de grafos, estas podem variar desde métricas topológicas simples, como a contagem de vértices e arestas [17], até descritores estatísticos mais complexos da estrutura do grafo. A eficácia de todo o sistema de seleção depende da qualidade destas características

em capturar os aspetos da estrutura do problema que influenciam o desempenho relativo dos algoritmos no Espaço de Algoritmos. O objetivo final, portanto, é construir o mapeamento de seleção

S(f(x)), que é, na sua essência, um modelo preditivo. Este modelo toma como entrada as características de uma nova instância de problema e produz como saída uma recomendação do algoritmo que se espera que otimize uma métrica no Espaço de Desempenho. Este enquadramento estabelece as bases para as abordagens de aprendizagem de máquina que serão discutidas na secção seguinte.

Secção 2: Seleção Baseada em Características e a Ascensão da Meta-Aprendizagem

Esta secção explora o "como" da seleção de algoritmos, focando-se no paradigma dominante de utilizar características de instâncias para treinar modelos de aprendizagem de máquina que preveem o desempenho.

2.1 A Seleção de Algoritmos como um Problema de Meta-Aprendizagem

O enquadramento de Rice transforma naturalmente o Problema de Seleção de Algoritmos numa tarefa de aprendizagem supervisionada, um campo conhecido como meta-aprendizagem.[13] A premissa central é aprender um modelo (o meta-modelo) que captura a relação entre as características de um problema e o desempenho dos algoritmos candidatos. Este processo começa com a construção de um meta-conjunto de dados. Cada ponto de dados neste conjunto corresponde a uma única instância de problema do Espaço de Problemas. As variáveis preditoras (features) são as características da instância, extraídas do Espaço de Características. A variável alvo (label) pode ser o identificador do algoritmo com melhor desempenho para essa instância (uma formulação de classificação) ou um vetor com as métricas de desempenho de todos os algoritmos (uma formulação de regressão).

Uma vez construído o meta-conjunto de dados, qualquer técnica de aprendizagem supervisionada pode ser aplicada para treinar o meta-modelo. Este modelo, uma vez treinado, pode ser usado para prever o algoritmo mais adequado para uma nova instância de problema, nunca antes vista, com base apenas nas suas características. Este processo automatiza a seleção, substituindo a dispendiosa experimentação manual ou a intuição do

especialista por uma decisão baseada em dados.

2.2 Caracterização do Problema: Análise Exploratória da Paisagem (ELA)

O sucesso de um seletor de algoritmos baseado em características depende fundamentalmente da "informatividade" do seu conjunto de características. A Análise Exploratória da Paisagem (ELA, do inglês *Exploratory Landscape Analysis*) surgiu como uma metodologia sistemática para calcular características numéricas que descrevem a paisagem de aptidão (fitness landscape) de um problema de otimização.[26] Em vez de se basear apenas em descritores de alto nível e específicos do domínio, a ELA utiliza técnicas de amostragem de baixo custo para sondar a paisagem de busca e extrair características quantitativas. Estas características servem como uma entrada rica e padronizada para o meta-modelo.

As características da ELA podem ser agrupadas em várias categorias, cada uma capturando um aspeto diferente da dificuldade do problema:

- **Características de Distribuição-y**: Medidas estatísticas da distribuição dos valores da função objetivo obtidos a partir de uma amostra de soluções (por exemplo, assimetria, curtose).[28]
- **Características de Conjunto de Nível (Levelset)**: Propriedades das soluções agrupadas por intervalos de valores da função objetivo.
- **Características de Meta-Modelo**: Características extraídas de um modelo substituto simples (por exemplo, um modelo linear) ajustado aos pontos de amostra. A qualidade do ajuste (por exemplo, R2) pode indicar a linearidade da paisagem.[28]
- **Características de Correlação Fitness-Distância**: Medem a correlação entre a aptidão de uma solução e a sua distância a outras soluções de alta qualidade, o que pode indicar a presença de "funis" de atração para ótimos globais.

Apesar da sua utilidade, a ELA tradicional enfrenta desafios, como a alta correlação entre muitas das suas características e a sua aplicabilidade limitada a certos tipos de problemas, como os multiobjetivo.[29] Para superar estas limitações, a investigação mais recente tem-se focado em abordagens de aprendizagem de características. Um exemplo proeminente é a Neural ELA (NeurELA), que utiliza redes neuronais, como transformadores baseados em atenção, para aprender representações de características de forma totalmente

end-to-end.[30] Este avanço representa uma mudança de paradigma, movendo a tarefa de engenharia de características do especialista humano para o próprio processo de aprendizagem, com o objetivo de descobrir representações mais poderosas e generalizáveis

da dificuldade do problema.

2.3 Seleção de Características na Meta-Aprendizagem

A riqueza dos conjuntos de características da ELA pode levar à "maldição da dimensionalidade" no nível da meta-aprendizagem. Um grande número de características pode ser redundante (altamente correlacionado) ou irrelevante para a tarefa de seleção, o que pode prejudicar o desempenho do meta-modelo, aumentar o custo computacional e diminuir a interpretabilidade.[31] Consequentemente, a seleção de características torna-se um passo crucial na construção de seletores de algoritmos robustos.

As técnicas padrão de seleção de características são diretamente aplicáveis neste contexto:

- **Métodos de Filtro (Filter Methods)**: Avaliam as características com base nas suas propriedades estatísticas intrínsecas, independentemente do meta-modelo. Por exemplo, pode-se usar a correlação de Pearson para remover características da ELA que são altamente correlacionadas entre si antes de treinar o seletor.[34]
- **Métodos de Invólucro (Wrapper Methods)**: Utilizam o próprio meta-modelo para avaliar a utilidade de subconjuntos de características. Um exemplo é a Eliminação Recursiva de Características (RFE), que treina iterativamente o meta-modelo e remove as características menos importantes.[32]
- **Métodos Embutidos (Embedded Methods)**: A seleção de características é integrada no processo de treino do meta-modelo. Modelos baseados em regularização, como LASSO, ou modelos baseados em árvores, como Random Forests, realizam a seleção de características implicitamente durante o treino.[32]

A aplicação destas técnicas resulta num meta-modelo mais parcimonioso, que não só tem maior probabilidade de generalizar para novos problemas, mas também é mais fácil de interpretar, um tópico que será explorado em detalhe na Secção 4.

Secção 3: Alocação Dinâmica de Recursos com Portfólios de Algoritmos 'Anytime'

Esta secção avança para além do paradigma estático "selecionar-um-e-executar" para a abordagem mais sofisticada de gerir um portfólio de algoritmos de forma dinâmica e online.

3.1 Do Melhor Único para um Portfólio de Especialistas

A abordagem de seleção baseada em características descrita na secção anterior baseia-se na premissa de que um modelo preditivo pode identificar o melhor algoritmo para uma dada instância *a priori*. No entanto, mesmo o seletor mais preciso pode cometer erros. As características da instância podem ser enganadoras, o modelo preditivo pode ser impreciso, ou o desempenho de algoritmos estocásticos pode exibir uma variância elevada, tornando uma única execução não representativa do seu desempenho típico.[37] Uma abordagem de portfólio de algoritmos mitiga este risco ao não depender de um único algoritmo. Em vez disso, aproveita as forças complementares de múltiplos algoritmos executados concorrentemente.[38]

É crucial distinguir entre portfólios estáticos e dinâmicos. Um **portfólio estático** envolve a execução de um conjunto fixo de algoritmos em paralelo, muitas vezes com uma alocação de recursos pré-definida (por exemplo, fatias de tempo iguais), até que um deles encontre uma solução.[38] Embora mais robusto do que a seleção de um único algoritmo, este método pode ser ineficiente, desperdiçando recursos em algoritmos que têm um desempenho fraco na instância atual. Em contraste, um

portfólio dinâmico adapta a sua estratégia de alocação de recursos durante a execução. Ele monitoriza o progresso dos algoritmos em tempo real e realoca dinamicamente os recursos computacionais para os candidatos mais promissores.[40] Esta abordagem adaptativa é fundamental para a eficiência em cenários com incerteza.

Tabela 2: Comparação de Portfólios de Algoritmos Estáticos e Dinâmicos

Característica	Portfólio Estático	Portfólio Dinâmico/'Anytime'
Tempo de Decisão	Apenas no início (offline)	Contínuo, durante a execução (online)
Alocação de Recursos	Fixa e pré-determinada	Adaptativa e baseada em feedback
Mecanismo de Feedback	Nenhum (baseado em previsão offline)	Monitorização do desempenho em tempo real

Objetivo Principal	Encontrar a solução final o mais rápido possível	Maximizar a qualidade da solução em qualquer ponto no tempo
Desafio Principal	Seleção inicial do portfólio	Equilíbrio exploração-exploração
Modelo Exemplo	Execução paralela com fatias de tempo iguais	Política baseada em Multi-Armed Bandit

3.2 A Propriedade 'Anytime': Desempenho como uma Função do Tempo

O conceito de portfólios dinâmicos está intrinsecamente ligado a algoritmos 'anytime'. Um algoritmo 'anytime' é aquele que pode ser interrompido a qualquer momento e fornecer uma solução válida, cuja qualidade se espera que melhore com tempo de execução adicional.[41] Muitas meta-heurísticas usadas para problemas de otimização combinatória, como o Recozimento Simulado e os Algoritmos Genéticos para particionamento de grafos, exibem esta propriedade. A qualidade da partição (medida, por exemplo, pelo tamanho do corte) melhora progressivamente à medida que o algoritmo explora o espaço de busca.

A propriedade 'anytime' provoca uma mudança fundamental no objetivo da otimização. Em vez de se focar apenas em encontrar a melhor solução final, o objetivo passa a ser otimizar a **trajetória de desempenho**, ou seja, maximizar a qualidade da solução disponível em *qualquer ponto no tempo*. Isto é particularmente valioso em aplicações do mundo real onde os orçamentos de tempo são incertos ou podem ser interrompidos prematuramente. O objetivo não é apenas a qualidade final, mas a qualidade como uma função do tempo.

3.3 Modelando o Problema de Seleção Online

A tarefa de alocar dinamicamente recursos a um portfólio de algoritmos 'anytime' pode ser elegantemente enquadrada como um problema de múltiplos bandidos (Multi-Armed Bandit).[37] Neste análogo, cada algoritmo no portfólio é um "braço" de uma slot machine. "Puxar um braço" corresponde a alocar uma fatia de tempo computacional a um algoritmo. A

"recompensa" é a melhoria no desempenho (por exemplo, redução no tamanho do corte) que o algoritmo proporciona nessa fatia de tempo.

O desafio central neste enquadramento é o clássico dilema de exploração-exploração (*exploration-exploitation trade-off*).[37] O seletor online deve decidir se deve

explorar, alocando tempo a um algoritmo com desempenho atualmente inferior na esperança de que possa superar os outros mais tarde, ou **explorar**, alocando tempo ao algoritmo que está atualmente a liderar para capitalizar o seu sucesso imediato.

Técnicas de aprendizagem online, como as políticas UCB (Upper Confidence Bound) ou Thompson Sampling, são usadas para gerir este dilema. Estas políticas mantêm um modelo do desempenho esperado de cada algoritmo e atualizam-no com base no feedback em tempo real. A cada passo de decisão, a política usa este modelo para calcular as prioridades (ou seja, as fatias de tempo) para a próxima iteração, equilibrando a exploração de algoritmos incertos com a exploração de algoritmos comprovadamente bons.[37]

Uma estratégia prática que emerge deste enquadramento é a de "sondar e comprometer" (*probe and commit*). Uma fase inicial de "sondagem" envolve a execução de todos os algoritmos do portfólio por um curto período de tempo para recolher dados de desempenho iniciais. Com base nos resultados desta sondagem, uma fase de "compromisso" aloca o restante orçamento de tempo ao algoritmo (ou subconjunto de algoritmos) que se mostrou mais promissor.[43] Esta abordagem representa uma forma simples mas eficaz de implementar o equilíbrio exploração-exploração. A transição para portfólios 'anytime' representa uma mudança fundamental na definição do problema: de otimizar o

desempenho final para otimizar a *trajetória de desempenho*. Isto requer uma passagem de modelos preditivos estáticos para sistemas de controlo adaptativos e dinâmicos.

Secção 4: Da Previsão à Compreensão: Aprendizagem de Máquina Interpretável para a Seleção de Algoritmos

Esta secção aborda a necessidade de transparência, passando de seletores "caixa-preta" para modelos que fornecem explicações compreensíveis para as suas recomendações.

4.1 A Necessidade de Interpretabilidade

Num contexto científico e de engenharia de alta criticidade, a precisão preditiva por si só é frequentemente insuficiente. Se um seletor de algoritmos, implementado como uma rede neuronal profunda, recomenda o Algoritmo A em vez do Algoritmo B, a questão natural que se segue é: "Porquê?". Sem uma resposta a esta pergunta, é difícil construir confiança no sistema, depurar falhas e, mais importante, gerar novo conhecimento científico.[44] A aprendizagem de máquina interpretável (iML, do inglês

interpretable machine learning) visa abrir a "caixa-preta" dos modelos de ML, tornando as suas decisões inteligíveis para os utilizadores humanos.

Quando aplicado à seleção de algoritmos, um modelo interpretável transcende o seu papel como uma ferramenta de automação para se tornar um instrumento de descoberta científica. Pode revelar as relações subjacentes entre as características do problema e o desempenho do algoritmo, gerando efetivamente hipóteses sobre o que torna um problema difícil ou fácil para uma determinada abordagem algorítmica. Por exemplo, pode ajudar a descobrir que um determinado tipo de heurística de melhoria local é particularmente eficaz para grafos com uma alta densidade de arestas, mas falha em grafos esparsos.

4.2 Árvores de Decisão como Modelos Interpretáveis

As Árvores de Classificação e Regressão (CART, do inglês *Classification and Regression Trees*) são uma classe de modelos de aprendizagem de máquina que são poderosos, não paramétricos e, crucialmente, inerentemente interpretáveis.[49] Uma árvore de decisão treinada pode ser diretamente traduzida num conjunto de regras "SE-ENTÃO" (

IF-THEN) que são legíveis por humanos.

O processo de extração de regras envolve a travessia da árvore desde o nó raiz até cada um dos nós folha:

- **Nós Internos**: Cada nó interno na árvore representa uma decisão baseada numa única característica da instância. No contexto da seleção de algoritmos para particionamento de grafos, um nó pode testar uma condição como SE densidade_do_grafo < 0.1.
- **Ramos**: Os ramos que emanam de um nó representam os resultados possíveis do teste (por exemplo, verdadeiro ou falso). Seguir um ramo corresponde a satisfazer a condição do nó.
- **Folhas (Nós Terminais)**: Cada nó folha representa a recomendação final do algoritmo.

Atingir um nó folha significa que foi satisfeita uma sequência de condições que leva a uma recomendação específica, como ENTÃO usar_Recozimento_Simulado.

Por exemplo, considere uma árvore de decisão hipotética treinada para selecionar um algoritmo de particionamento de grafos. Uma travessia de um caminho da raiz a uma folha poderia gerar a seguinte regra: SE |V| > 10000 E densidade < 0.01 E distribuição_de_graus ~ lei_de_potência ENTÃO selecionar_Algoritmo_Multinível. Esta regra não é apenas uma recomendação; é uma hipótese explícita sobre a interação entre as propriedades do grafo e o desempenho do algoritmo, que pode ser validada e compreendida por um especialista no domínio.

4.3 Vantagens da Abordagem Interpretável

A utilização de modelos interpretáveis como as árvores de decisão para a seleção de algoritmos oferece várias vantagens significativas:

- **Transparência e Auditoria**: A lógica de decisão é explícita e pode ser facilmente inspecionada e validada por um especialista humano. Isto é crucial para a depuração e para garantir que o modelo não aprendeu correlações espúrias.[52]
- **Geração de Conhecimento**: As próprias regras extraídas constituem novos conhecimentos. Podem confirmar a sabedoria convencional (por exemplo, que o Recozimento Simulado funciona bem em grafos esparsos e uniformes [19]) ou revelar novas e inesperadas relações entre as características do problema e o desempenho do algoritmo.
- **Depuração Simplificada**: Se o seletor fizer uma recomendação incorreta, o caminho de decisão que levou a essa recomendação pode ser rastreado. Isto permite identificar se o erro se deve a uma característica mal calculada, a uma regra incorreta na árvore ou a uma região do espaço de problemas que não foi adequadamente representada nos dados de treino.
- **Seleção de Características Implícita**: O algoritmo de construção de árvores CART realiza inerentemente a seleção de características. Apenas as características mais discriminativas são escolhidas para os nós de divisão, o que significa que a árvore final destaca automaticamente as propriedades mais importantes da instância para a tarefa de seleção.[50]

Em suma, a aplicação da iML transforma o seletor de algoritmos de uma mera ferramenta de engenharia que fornece uma recomendação, para um instrumento científico que oferece uma explicação. Esta capacidade eleva o seu papel da automação para a descoberta de conhecimento, permitindo-nos passar da *previsão* (o que funcionará?) para a *explicação* (porque é que funcionará?) e, finalmente, para a *compreensão* (quais são os princípios

fundamentais que governam o desempenho do algoritmo nesta classe de problemas?).

Secção 5: Uma Estrutura Rigorosa para a Avaliação Experimental

Esta secção detalha as metodologias necessárias para validar empiricamente as afirmações feitas pelos sistemas de seleção de algoritmos, garantindo que as comparações sejam justas, robustas e reproduzíveis.

5.1 Desenho de Benchmarks: A Importância de Instâncias de Problemas Diversificadas

A credibilidade de qualquer estudo de seleção de algoritmos depende da qualidade e diversidade do conjunto de problemas de teste (benchmark). Um benchmark robusto deve incluir tanto instâncias do mundo real, para garantir a relevância prática, como instâncias sintéticas, para permitir experiências controladas e análise de escalabilidade.

- **Benchmarks do Mundo Real**: A utilização de coleções de grafos do mundo real é essencial para garantir que um seletor de algoritmos é eficaz em cenários práticos. Coleções como as dos desafios DIMACS [16] e benchmarks de nível industrial como o LDBC Graphalytics [17] fornecem conjuntos de dados padronizados que representam uma variedade de domínios, como redes sociais, conhecimento e jogos. Estes benchmarks são cruciais para avaliar o desempenho em grafos com as complexidades estruturais encontradas em aplicações reais.
- **Benchmarks Sintéticos**: Geradores de grafos sintéticos desempenham um papel complementar, permitindo aos investigadores controlar sistematicamente as propriedades do grafo (por exemplo, tamanho, densidade, estrutura de comunidade) para testar hipóteses específicas sobre o comportamento do algoritmo.
    - **Modelos para Estrutura de Comunidade**: Para problemas como a deteção de comunidades (uma forma de particionamento de grafos), modelos como o LFR (Lancichinetti-Fortunato-Radicchi) e o ABCD (Artificial Benchmark for Community Detection) são fundamentais. O modelo LFR é conhecido por gerar grafos com distribuições de grau e de tamanho de comunidade do tipo lei de potência, imitando muitas redes do mundo real.[54] No entanto, o modelo ABCD mais recente oferece vantagens significativas em termos de velocidade de geração, simplicidade teórica e

um parâmetro de mistura mais intuitivo, tornando-o uma alternativa poderosa.[57] O Modelo de Blocos Estocásticos (SBM, do inglês *Stochastic Block Model*), e a sua variante "partição plantada", é outro modelo gerador fundamental que cria grafos com uma estrutura de comunidade de base conhecida, servindo como um benchmark ideal para avaliar a precisão dos algoritmos de recuperação de comunidades.[58]

- **Armadilhas na Avaliação**: É imperativo estar ciente das armadilhas comuns na avaliação experimental. Metodologias falhas, como a validação cruzada "deixar-instância-de-fora" (*leave-instance-out*) em benchmarks onde as instâncias dentro da mesma classe de problema são altamente semelhantes, podem levar a resultados excessivamente otimistas e enganadores. Da mesma forma, o uso de métricas de desempenho sensíveis à escala (por exemplo, erro absoluto em vez de erro relativo) pode distorcer as comparações e favorecer meta-modelos que simplesmente aprendem a escala do problema em vez da sua dificuldade intrínseca.[62]

5.2 Análise Estatística: Tirando Conclusões Sólidas

Uma vez que os dados de desempenho são recolhidos, é necessária uma análise estatística rigorosa para determinar se as diferenças observadas são significativas ou meramente resultado do acaso, especialmente ao lidar com algoritmos estocásticos. Como argumentado por Demšar (2006), as suposições subjacentes aos testes paramétricos (como o teste t ou a ANOVA) são frequentemente violadas em estudos de comparação de algoritmos. As métricas de desempenho raramente seguem uma distribuição normal, e as variâncias não são homogéneas. Portanto, os testes não paramétricos são considerados mais seguros e robustos para esta tarefa.[67]

- **Comparação de Dois Algoritmos**: Para comparar o desempenho de dois algoritmos em múltiplos conjuntos de dados, o **teste de postos com sinal de Wilcoxon** é a ferramenta recomendada. É a alternativa não paramétrica ao teste t pareado e avalia se as diferenças medianas entre os desempenhos dos dois algoritmos são significativamente diferentes de zero.[67]
- **Comparação de Múltiplos Algoritmos**: Quando se comparam três ou mais algoritmos, é necessário um procedimento de dois passos para evitar o problema das comparações múltiplas.
  1. **Teste Omnibus**: O **teste de Friedman** é usado primeiro. É a contrapartida não paramétrica da ANOVA de medidas repetidas. O teste de Friedman classifica o desempenho dos algoritmos em cada conjunto de dados e depois testa a hipótese nula de que todos os algoritmos têm um desempenho equivalente (ou seja, os seus postos médios são os mesmos).[67]

2. **Análise Post-Hoc**: Se o teste de Friedman rejeitar a hipótese nula (indicando que existe uma diferença significativa entre pelo menos dois algoritmos), um teste post-hoc é realizado para identificar quais pares de algoritmos diferem significativamente. O **teste de Nemenyi** é o procedimento post-hoc padrão para comparações par a par após um teste de Friedman significativo. Ele calcula uma "diferença crítica" e considera que quaisquer dois algoritmos cujos postos médios difiram por mais do que este valor têm um desempenho estatisticamente diferente.[67]

- **Medição do Tamanho do Efeito**: Os testes de significância indicam se existe uma diferença, mas não a sua magnitude. A estatística **A12 de Vargha-Delaney** é uma medida de tamanho de efeito não paramétrica que complementa os testes de hipóteses. A A12 quantifica a probabilidade de um algoritmo A obter um resultado melhor do que um algoritmo B, se uma execução de cada um for escolhida aleatoriamente. Um valor de A12 de 0.7 significa que A superará B em 70% das vezes. Isto fornece uma interpretação intuitiva e prática da magnitude da diferença de desempenho.[77]

Tabela 3: Um Guia Prático para Testes Estatísticos na Comparação de Algoritmos

Nome do Teste	Caso de Uso	Hipótese Nula (H_0)	Interpretação da Rejeição
Teste de Postos com Sinal de Wilcoxon	Comparar 2 algoritmos em múltiplos conjuntos de dados.	A diferença mediana no desempenho entre os dois algoritmos é zero.	Existe uma diferença estatisticamente significativa no desempenho entre os dois algoritmos.
Teste de Friedman	Comparar 3+ algoritmos em múltiplos conjuntos de dados.	Todos os algoritmos têm um desempenho equivalente (os seus postos médios são iguais).	Existe uma diferença estatisticamente significativa no desempenho de pelo menos um algoritmo em comparação com os outros.
Teste Post-Hoc de Nemenyi	Comparações par a par após um teste de Friedman significativo.	O desempenho de dois algoritmos específicos é equivalente (os	O desempenho dos dois algoritmos comparados é estatisticamente

		seus postos médios não diferem significativamente).	diferente.

5.3 Visualização do Desempenho: Para Além das Tabelas de Números

Enquanto os testes estatísticos fornecem validação, as visualizações oferecem uma compreensão intuitiva dos compromissos de desempenho.

- **Perfis de Desempenho (Dolan & Moré)**: Os perfis de desempenho são uma ferramenta padrão para comparar múltiplos solvers num conjunto de problemas.[80] Para cada problema
  p e cada solver s, calcula-se o rácio de desempenho $r_{p,s}$, que é o custo do solver s (por exemplo, tempo de CPU) dividido pelo custo do melhor solver para esse problema. O perfil de desempenho para um solver s, $\rho_s(\tau)$, é então a fração de problemas para os quais o rácio de desempenho do solver é no máximo τ. A interpretação do gráfico é direta:
  - **Eficiência**: O valor de $\rho_s(1)$ no eixo y representa a fração de problemas para os quais o solver s foi o melhor (ou um dos melhores). Um valor mais alto indica maior eficiência.
  - **Robustez**: A assíntota direita do gráfico (o valor de $\rho_s(\tau)$ para τ grande) representa a fração total de problemas que o solver s conseguiu resolver. Um valor mais alto indica maior robustez.[82]
- **Gráficos de Tempo para o Alvo (TTT) / Gráficos ECDF**: Para algoritmos estocásticos, é crucial visualizar a distribuição dos tempos de execução. Os gráficos de Tempo para o Alvo (TTT), que são uma forma de Função de Distribuição Cumulativa Empírica (ECDF), são ideais para este fim.[89] O gráfico mostra a probabilidade (eixo y) de um algoritmo atingir uma solução de uma determinada qualidade alvo dentro de um dado tempo de execução (eixo x). Um aspeto crítico e frequentemente mal compreendido destes gráficos é o tratamento de execuções mal sucedidas. As execuções que não atingem o valor alvo dentro de um tempo limite pré-definido são tratadas como **dados censurados**. A altura final da curva ECDF (o seu ponto de saturação) representa diretamente a **taxa de sucesso**, ou seja, a fração de execuções que conseguiram atingir o alvo. Uma curva que satura a 0.8 indica que o algoritmo teve sucesso em 80% das execuções.[92]

Um benchmark rigoroso é, portanto, um sistema tripartido que consiste em (1) problemas diversos e representativos, (2) validação estatística robusta e (3) visualizações intuitivas e

informativas. A fraqueza em qualquer um destes pilares compromete toda a avaliação.

Secção 6: Melhores Práticas para a Investigação Reproduzível na Seleção de Algoritmos

Esta secção final fornece um guia prático para a implementação dos princípios da investigação reproduzível no contexto de um projeto de seleção de algoritmos, ligando conceitos de alto nível a uma cadeia de ferramentas específica e moderna.

6.1 Os Cinco Pilares da Reprodutibilidade Computacional

A reprodutibilidade não é um luxo, mas um pilar fundamental do método científico. A capacidade de outros investigadores replicarem, validarem e estenderem os resultados é essencial para o progresso científico. A estrutura dos "Cinco Pilares da Reprodutibilidade Computacional" oferece um guia abrangente para a organização de projetos de investigação computacional.[108] Estes pilares são:

1. **Programação Literária (Literate Programming)**: Integrar código, texto e resultados num único documento (por exemplo, Jupyter Notebooks, R Markdown) para criar uma narrativa computacional que documente a proveniência de cada resultado.
2. **Controlo de Versão e Partilha de Código (Code Version Control and Sharing)**: Utilizar sistemas de controlo de versão como o Git para rastrear todas as alterações ao código e partilhá-lo em repositórios públicos.
3. **Controlo do Ambiente Computacional (Compute Environment Control)**: Capturar e partilhar a configuração exata do ambiente de software (bibliotecas, versões, dependências) para garantir que o código é executado da mesma forma em diferentes máquinas.
4. **Partilha Persistente de Dados (Persistent Data Sharing)**: Assegurar que os dados brutos e processados são armazenados de forma segura e acessível a longo prazo, utilizando repositórios de dados dedicados.
5. **Documentação (Documentation)**: Fornecer documentação clara e abrangente que explique a estrutura do projeto, os passos da análise e como reproduzir os resultados.

Esta estrutura moderna e abrangente expande as "Dez Regras Simples" mais antigas, mas ainda relevantes, para a investigação computacional reproduzível.[54]

6.2 Uma Cadeia de Ferramentas Prática para a Reprodutibilidade

A verdadeira reprodutibilidade é um princípio de design que deve ser arquitetado num projeto de investigação desde o seu início. Um fluxo de trabalho moderno e reproduzível é um ecossistema coeso onde o controlo de versões, a gestão de ambientes e as ferramentas de automação de fluxos de trabalho funcionam em conjunto.

- **Controlo de Versão de Código e Dados (Pilares 2 & 4)**:
  - **Git**: É a ferramenta padrão para o controlo de versões de código-fonte (scripts, Makefiles, ficheiros de configuração). A sua natureza distribuída facilita a colaboração e a manutenção de um histórico completo de todas as alterações.[109]
  - **Git LFS vs. DVC**: Ficheiros grandes, como conjuntos de dados e modelos de ML, não devem ser armazenados diretamente em repositórios Git. O Git LFS (Large File Storage) é uma extensão que substitui ficheiros grandes por ponteiros de texto, mas carece de funcionalidades específicas para ML, como a consciência de pipelines.[114] O DVC (Data Version Control) é uma ferramenta construída especificamente para fluxos de trabalho de ML. Funciona em conjunto com o Git para versionar dados e modelos, armazenando-os em armazenamento remoto (por exemplo, S3, GCS) e ligando-os a commits específicos do Git. As suas capacidades de gestão de pipelines e rastreio de experiências tornam-no a escolha superior para investigação séria em ML.[116]
- **Controlo do Ambiente Computacional (Pilar 3)**:
  - **Conda vs. Poetry**: A gestão de dependências de software é crítica. O **Conda** é uma ferramenta poderosa que gere tanto pacotes Python como não-Python, o que é comum em computação científica. Permite a criação de ambientes isolados e reproduzíveis definidos num ficheiro environment.yml.[119] O **Poetry** é uma ferramenta mais moderna e focada no ecossistema Python, que oferece uma resolução de dependências mais determinística e uma gestão de projetos simplificada através de um ficheiro pyproject.toml.[119] Para projetos que dependem de bibliotecas de sistemas complexas, uma abordagem híbrida, utilizando o Conda para gerir o ambiente base e o Poetry para as dependências Python dentro desse ambiente, pode oferecer o melhor de ambos os mundos.[122]
- **Automação de Fluxos de Trabalho (Ligando todos os Pilares)**:
  - **Make**: A ferramenta Make, com o seu ficheiro de configuração Makefile, é uma forma robusta e testada pelo tempo de automatizar fluxos de trabalho computacionais. Um Makefile define um grafo de dependências entre ficheiros: os resultados (alvos) dependem de dados e scripts (pré-requisitos). O Make verifica os carimbos de data/hora e reexecuta apenas os passos necessários quando um pré-requisito é modificado. Isto não só poupa tempo computacional, mas também garante que o

fluxo de trabalho é executado na ordem correta, desde o pré-processamento dos dados até à geração de figuras e do relatório final, tornando todo o processo totalmente automatizado e reproduzível.[124]

- **Validação e Integridade de Dados**:
  - **JSON Schema**: Para garantir a integridade dos dados que fluem através de um pipeline (por exemplo, ficheiros de configuração, resultados intermédios), o JSON Schema pode ser usado para definir e validar a sua estrutura. Ao definir um esquema formal para os ficheiros JSON, pode-se verificar programaticamente se os dados estão em conformidade com o formato esperado, prevenindo erros causados por dados malformados e melhorando a robustez e reprodutibilidade do pipeline.[129]

Ao integrar estas ferramentas, um projeto de investigação em seleção de algoritmos passa de um conjunto de scripts ad-hoc para um objeto de investigação auto-contido, auto-validado e totalmente automatizado. Este objeto encarna os princípios dos Cinco Pilares, representando uma transição da programação artesanal para a engenharia computacional rigorosa, que é o padrão-ouro da investigação científica moderna.

Conclusões

A seleção de algoritmos evoluiu de um problema heurístico para um campo de estudo rigoroso, sustentado por fundamentos teóricos e impulsionado por metodologias sofisticadas de aprendizagem de máquina e engenharia de software. Esta análise abrangente traçou a trajetória deste campo, desde os seus princípios teóricos até às suas implementações práticas mais avançadas, culminando numa visão para o futuro da investigação algorítmica.

A jornada começa com o reconhecimento, formalizado pelos teoremas "No Free Lunch", de que não existe um algoritmo universalmente ótimo. Esta constatação teórica não é uma barreira, mas sim a principal justificação para a seleção de algoritmos: precisamente porque os problemas do mundo real possuem estrutura, existe a oportunidade de explorar essa estrutura para obter um desempenho superior. O modelo seminal de Rice forneceu a linguagem para formalizar esta busca, decompondo o problema nos espaços de problemas, características, algoritmos e desempenho.

A operacionalização deste modelo deu origem à meta-aprendizagem, onde a seleção de algoritmos é enquadrada como uma tarefa de aprendizagem supervisionada. O sucesso desta abordagem depende da capacidade de caracterizar numericamente a dificuldade de uma instância de problema, uma tarefa para a qual a Análise Exploratória da Paisagem (ELA) oferece um conjunto sistemático de ferramentas. A evolução da engenharia de características, desde descritores simples até características ELA complexas e, mais recentemente, representações aprendidas de ponta a ponta, reflete uma busca contínua por

uma linguagem mais expressiva para descrever a interação entre problemas e algoritmos.

No entanto, a seleção estática, baseada em previsões a priori, tem limitações inerentes face à incerteza e à natureza estocástica de muitas meta-heurísticas. A transição para portfólios de algoritmos 'anytime' representa uma mudança de paradigma fundamental. Ao reformular o problema como uma tarefa de alocação de recursos online, gerida por políticas adaptativas inspiradas em problemas de múltiplos bandidos, o campo passou da previsão estática para o controlo dinâmico. O objetivo já não é apenas otimizar o resultado final, mas sim a trajetória de desempenho ao longo do tempo, uma abordagem mais robusta e relevante para aplicações com restrições de tempo variáveis.

Além da precisão preditiva, a necessidade de compreensão impulsionou a integração da aprendizagem de máquina interpretável. A utilização de modelos como as árvores de decisão transforma o seletor de algoritmos de uma "caixa-preta" numa ferramenta de descoberta de conhecimento. As regras "SE-ENTÃO" extraídas destes modelos não só explicam as recomendações, mas também geram hipóteses testáveis sobre o que torna um problema difícil para um determinado algoritmo, promovendo um ciclo virtuoso entre a automação e a compreensão científica.

Finalmente, a validade de todas estas abordagens depende de uma avaliação experimental rigorosa e reproduzível. Um enquadramento de benchmarking robusto requer uma combinação de conjuntos de dados do mundo real e sintéticos, a aplicação de testes estatísticos não paramétricos adequados para evitar conclusões espúrias, e o uso de ferramentas de visualização como os Perfis de Desempenho e os gráficos de Tempo para o Alvo para comunicar os resultados de forma clara e honesta. A adesão a práticas de investigação reproduzível, encapsuladas nos Cinco Pilares e implementadas através de uma cadeia de ferramentas moderna (como Git, DVC, Poetry, Make e JSON Schema), é o que garante que os avanços no campo são cumulativos, verificáveis e fiáveis.

Em suma, o Problema de Seleção de Algoritmos está no cerne da ciência da computação moderna. A sua resolução eficaz exige uma síntese de teoria da otimização, aprendizagem de máquina, estatística e engenharia de software. O futuro do campo reside na continuação da integração destas disciplinas: no desenvolvimento de seletores mais adaptativos e online, na extração de conhecimento mais profundo através de modelos interpretáveis e na adesão a padrões cada vez mais elevados de rigor metodológico e reprodutibilidade.

Referências citadas

1. No free lunch in search and optimization - Wikipedia, acessado em agosto 29, 2025, https://en.wikipedia.org/wiki/No_free_lunch_in_search_and_optimization
2. No Free Lunch Theorem and Its Foundational Implications for Algorithm Selection in Artificial Intelligence | by Adnan Masood, PhD. - Medium, acessado em agosto 29, 2025, https://medium.com/@adnanmasood/no-free-lunch-theorem-and-its-foundation

al-implications-for-algorithm-selection-in-artificial-5fc49c218d76
3. Simple Explanation of the No-Free-Lunch Theorem and Its Implications, acessado em agosto 29, 2025, http://web.cecs.pdx.edu/~greenwd/NFL.pdf
4. What are the practical implications of "no free lunch" theorems for ..., acessado em agosto 29, 2025, https://www.researchgate.net/post/What-are-the-practical-implications-of-no-free-lunch-theorems-for-optimization
5. Are the No Free Lunch theorems useful for anything? - Computer Science Stack Exchange, acessado em agosto 29, 2025, https://cs.stackexchange.com/questions/88192/are-the-no-free-lunch-theorems-useful-for-anything
6. Partitioning Trillion-edge Graphs in Minutes - OSTI.GOV, acessado em agosto 29, 2025, https://www.osti.gov/servlets/purl/1484927
7. J. R. Rice. "The algorithm selection problem ". Advances in Computers, vol. 15, pp. 65-118, 1976., acessado em agosto 29, 2025, https://www.sciepub.com/reference/84716
8. The Algorithm Selection Problem - AMiner, acessado em agosto 29, 2025, https://www.cn.aminer.org/pub/53e99adcb7602d970235e723
9. Optimal decision trees for the Algorithm Selection Problem - TU Delft Repository, acessado em agosto 29, 2025, https://repository.tudelft.nl/file/File_2d4ccf86-6d8d-4a0d-b039-c2aceb1740db
10. The Algorithm Selection Problem (Rice, 1976) - ResearchGate, acessado em agosto 29, 2025, https://www.researchgate.net/figure/The-Algorithm-Selection-Problem-Rice-1976_fig1_335995943
11. The Algorithm Selection Problem—Abstract Models - Purdue e-Pubs, acessado em agosto 29, 2025, https://docs.lib.purdue.edu/cgi/viewcontent.cgi?article=1067&context=cstech
12. The Algorithm Selection Problem - Purdue e-Pubs, acessado em agosto 29, 2025, https://docs.lib.purdue.edu/cgi/viewcontent.cgi?article=1098&context=cstech
13. An Overview of the Algorithm Selection Problem - ResearchGate, acessado em agosto 29, 2025, https://www.researchgate.net/publication/319718715_An_Overview_of_the_Algorithm_Selection_Problem
14. View of An Overview of the Algorithm Selection Problem - International Journal of Computer, acessado em agosto 29, 2025, https://ijcjournal.org/InternationalJournalOfComputer/article/view/1016/446
15. (PDF) Cross-Disciplinary Perspectives on Meta-Learning for Algorithm Selection, acessado em agosto 29, 2025, https://www.researchgate.net/publication/220565856_Cross-Disciplinary_Perspectives_on_Meta-Learning_for_Algorithm_Selection
16. Graph Partitioning and Graph Clustering - David A. Bader, acessado em agosto 29, 2025, https://davidbader.net/publication/2013-bmsw/2013-bmsw.pdf
17. LDBC Graphalytics: A Benchmark for Large-Scale Graph Analysis ..., acessado em agosto 29, 2025, https://www.vldb.org/pvldb/vol9/p1317-iosup.pdf

18. Recent Advances in Graph Partitioning, acessado em agosto 29, 2025, https://www.eecis.udel.edu/~isafro/papers/advpart.pdf
19. A seed-growth heuristic for graph bisection - Harvard DASH, acessado em agosto 29, 2025, https://dash.harvard.edu/bitstreams/7312037c-49f0-6bd4-e053-0100007fdf3b/download
20. (PDF) Algorithms for Graph Partitioning: A Survey - ResearchGate, acessado em agosto 29, 2025, https://www.researchgate.net/publication/2828688_Algorithms_for_Graph_Partitioning_A_Survey
21. Selecting Machine Learning Algorithms Using the Ranking Meta-Learning Approach - CIn UFPE, acessado em agosto 29, 2025, https://www.cin.ufpe.br/~rbcp/papers/book_chapter-final-v6.pdf
22. (PDF) Meta-Learning and Algorithm Selection - ResearchGate, acessado em agosto 29, 2025, https://www.researchgate.net/publication/275966998_Meta-Learning_and_Algorithm_Selection
23. Survey on Meta-Learning Research of Algorithm Selection - SciEngine, acessado em agosto 29, 2025, https://www.sciengine.com/doi/10.3778/j.issn.1673-9418.2204019
24. [PDF] Automated Algorithm Selection: Survey and Perspectives - Semantic Scholar, acessado em agosto 29, 2025, https://www.semanticscholar.org/paper/Automated-Algorithm-Selection%3A-Survey-and-Kerschke-Hoos/be1fa570187f20c03b97fa0e9b09f44e73eb33c2
25. [Literature Review] An experimental survey and Perspective View on Meta-Learning for Automated Algorithms Selection and Parametrization - Moonlight, acessado em agosto 29, 2025, https://www.themoonlight.io/en/review/an-experimental-survey-and-perspective-view-on-meta-learning-for-automated-algorithms-selection-and-parametrization
26. Exploratory landscape analysis - Statistical Learning and Data Science, acessado em agosto 29, 2025, https://www.slds.stat.uni-muenchen.de/bibrefs/pdfs/exploratory_landscape_analysis.pdf
27. Exploratory Landscape Analysis (ELA) - COSEAL, acessado em agosto 29, 2025, https://www.coseal.net/ela/
28. Algorithm Selection Based on Exploratory Landscape Analysis and Cost-Sensitive Learning - CMAP, acessado em agosto 29, 2025, http://www.cmap.polytechnique.fr/~nikolaus.hansen/proceedings/2012/GECCO/proceedings/p313.pdf
29. www.researchgate.net, acessado em agosto 29, 2025, https://www.researchgate.net/publication/388683810_Deep-ELA_Deep_Exploratory_Landscape_Analysis_with_Self-Supervised_Pretrained_Transformers_for_Single-_and_Multi-Objective_Continuous_Optimization_Problems#:~:text=knowledge%E2%80%94very%20limited.-,Yet%2C%20despite%20their%20usefulness%2C%

20as%20demonstrated%20in%20several%20past%20works.to%20multiobjective%20continuous%20optimization%20problems.
30. Neural Exploratory Landscape Analysis for Meta-Black-Box-Optimization | OpenReview, acessado em agosto 29, 2025, https://openreview.net/forum?id=EEI5R89Cmv
31. Feature selection - Wikipedia, acessado em agosto 29, 2025, https://en.wikipedia.org/wiki/Feature_selection
32. Understanding Feature Selection Techniques in Machine Learning | by NIRAJAN JHA, acessado em agosto 29, 2025, https://medium.com/@nirajan_DataAnalyst/understanding-feature-selection-techniques-in-machine-learning-02e2642ef63e
33. What is Feature Selection? | IBM, acessado em agosto 29, 2025, https://www.ibm.com/think/topics/feature-selection
34. How to Choose a Feature Selection Method For Machine Learning - MachineLearningMastery.com, acessado em agosto 29, 2025, https://machinelearningmastery.com/feature-selection-with-real-and-categorical-data/
35. Feature Selection Techniques in Machine Learning - GeeksforGeeks, acessado em agosto 29, 2025, https://www.geeksforgeeks.org/machine-learning/feature-selection-techniques-in-machine-learning/
36. Introduction to Feature Selection - MATLAB & Simulink - MathWorks, acessado em agosto 29, 2025, https://www.mathworks.com/help/stats/feature-selection.html
37. Learning Dynamic Algorithm Portfolios - IDSIA, acessado em agosto 29, 2025, https://sferics.idsia.ch/pub/juergen/gagliolo-amai837.pdf
38. Algorithm Selection for Combinatorial Search Problems: A Survey - AAAI Publications, acessado em agosto 29, 2025, https://ojs.aaai.org/aimagazine/index.php/aimagazine/article/view/2460/2438
39. Algorithm portfolios | Request PDF - ResearchGate, acessado em agosto 29, 2025, https://www.researchgate.net/publication/222831786_Algorithm_portfolios
40. Algorithm Selection for Combinatorial Search Problems: A survey - Department of Electrical Engineering and Computer Science, acessado em agosto 29, 2025, https://www.cs.uwyo.edu/~larsko/papers/kotthoff_algorithm_2012-1.pdf
41. Algorithm selection of anytime algorithms | Request PDF - ResearchGate, acessado em agosto 29, 2025, https://www.researchgate.net/publication/342540437_Algorithm_selection_of_anytime_algorithms
42. Heuristics in dynamic scheduling: A practical framework with a case study in elevator dispatching - SciSpace, acessado em agosto 29, 2025, https://scispace.com/pdf/heuristics-in-dynamic-scheduling-a-practical-framework-with-4ljn6jg2ge.pdf
43. Dynamic Algorithm Portfolios - CiteSeerX, acessado em agosto 29, 2025, https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=fe2e3f7d950da045e4a390f0ce9ef5b1e313b911

44. Interpretable machine learning for genomics - PMC, acessado em agosto 29, 2025, https://pmc.ncbi.nlm.nih.gov/articles/PMC8527313/
45. Definitions, methods, and applications in interpretable machine learning - PNAS, acessado em agosto 29, 2025, https://www.pnas.org/doi/10.1073/pnas.1900654116
46. 2 Interpretability – Interpretable Machine Learning, acessado em agosto 29, 2025, https://christophm.github.io/interpretable-ml-book/interpretability.html
47. Interpretable Machine Learning - Christoph Molnar, acessado em agosto 29, 2025, https://christophm.github.io/interpretable-ml-book/
48. Guide to Interpretable Machine Learning | by Matthew Stewart, PhD | TDS Archive | Medium, acessado em agosto 29, 2025, https://medium.com/data-science/guide-to-interpretable-machine-learning-d40e8a64b6cf
49. FEATURE SELECTION BY USING CLASSIFICATION AND ..., acessado em agosto 29, 2025, https://www.isprs.org/proceedings/xxxv/congress/comm7/papers/13.pdf
50. CART (Classification And Regression Tree) in Machine Learning ..., acessado em agosto 29, 2025, https://www.geeksforgeeks.org/machine-learning/cart-classification-and-regression-tree-in-machine-learning/
51. Decision tree learning - Wikipedia, acessado em agosto 29, 2025, https://en.wikipedia.org/wiki/Decision_tree_learning
52. 1.10. Decision Trees - Scikit-learn, acessado em agosto 29, 2025, https://scikit-learn.org/stable/modules/tree.html
53. LDBC Graphalytics: A Benchmark for Large-Scale Graph Analysis on Parallel and Distributed Platforms - TU Delft Research Portal, acessado em agosto 29, 2025, https://research.tudelft.nl/files/66845401/p1317_iosup.pdf
54. Ten Simple Rules for Reproducible Computational Research - PMC, acessado em agosto 29, 2025, https://pmc.ncbi.nlm.nih.gov/articles/PMC3812051/
55. Lancichinetti–Fortunato–Radicchi benchmark - Wikipedia, acessado em agosto 29, 2025, https://en.wikipedia.org/wiki/Lancichinetti%E2%80%93Fortunato%E2%80%93Radicchi_benchmark
56. LFR_benchmark_graph — NetworkX 3.5 documentation, acessado em agosto 29, 2025, https://networkx.org/documentation/stable/reference/generated/networkx.generators.community.LFR_benchmark_graph.html
57. Artificial Benchmark for Community Detection (ABCD)—Fast random ..., acessado em agosto 29, 2025, https://www.cambridge.org/core/services/aop-cambridge-core/content/view/453FE29A1FA3C1798B0EC116587FE422/S2050124220000454a.pdf/artificial-benchmark-for-community-detection-abcdfast-random-graph-model-with-community-structure.pdf
58. Stochastic block model - Wikipedia, acessado em agosto 29, 2025, https://en.wikipedia.org/wiki/Stochastic_block_model
59. Improved Community Detection using Stochastic Block Models - arXiv, acessado

em agosto 29, 2025, https://arxiv.org/html/2408.10464v2
60. Structural Entropy of the Stochastic Block Models - MDPI, acessado em agosto 29, 2025, https://www.mdpi.com/1099-4300/24/1/81
61. SBGD: Improving Graph Diffusion Generative Model via Stochastic Block Diffusion - arXiv, acessado em agosto 29, 2025, https://arxiv.org/html/2508.14352v1
62. The Pitfalls of Benchmarking in Algorithm Selection: What We Are Getting Wrong - arXiv, acessado em agosto 29, 2025, https://arxiv.org/html/2505.07750v1
63. The Pitfalls of Benchmarking in Algorithm Selection: What We ... - arXiv, acessado em agosto 29, 2025, https://arxiv.org/pdf/2505.07750
64. The Pitfalls of Benchmarking in Algorithm Selection: What We Are Getting Wrong, acessado em agosto 29, 2025, https://www.researchgate.net/publication/393651527_The_Pitfalls_of_Benchmarking_in_Algorithm_Selection_What_We_Are_Getting_Wrong
65. Rethinking Metaheuristics: Unveiling the Myth of “Novelty” in Metaheuristic Algorithms, acessado em agosto 29, 2025, https://www.mdpi.com/2227-7390/13/13/2158
66. Metaheuristics exposed: Unmasking the design pitfalls of arithmetic optimization algorithm in benchmarking | Request PDF - ResearchGate, acessado em agosto 29, 2025, https://www.researchgate.net/publication/380380954_Metaheuristics_exposed_Unmasking_the_design_pitfalls_of_arithmetic_optimization_algorithm_in_benchmarking
67. Statistical Comparisons of Classifiers over Multiple Data Sets, acessado em agosto 29, 2025, https://www.jmlr.org/papers/volume7/demsar06a/demsar06a.pdf
68. (PDF) Statistical Comparisons of Classifiers over Multiple Data Sets (2006) | Janez Demšar | 12256 Citations - SciSpace, acessado em agosto 29, 2025, https://scispace.com/papers/statistical-comparisons-of-classifiers-over-multiple-data-18bm1e3eph
69. Understanding the Wilcoxon Sign Test - Statistics Solutions, acessado em agosto 29, 2025, https://www.statisticssolutions.com/free-resources/directory-of-statistical-analyses/wilcoxon-sign-test/
70. Wilcoxon signed-rank test - Wikipedia, acessado em agosto 29, 2025, https://en.wikipedia.org/wiki/Wilcoxon_signed-rank_test
71. Wilcoxon Signed Rank Test - GeeksforGeeks, acessado em agosto 29, 2025, https://www.geeksforgeeks.org/machine-learning/wilcoxon-signed-rank-test/
72. Statistics: 2.2 The Wilcoxon signed rank sum test - Statstutor, acessado em agosto 29, 2025, https://www.statstutor.ac.uk/resources/uploaded/wilcoxonsignedranktest.pdf
73. A Graphical Approach for Friedman Test: Moments Approach - arXiv, acessado em agosto 29, 2025, https://arxiv.org/pdf/2202.09131
74. How to Show that Your Model is Better: A Basic Guide to Statistical Hypothesis Testing, acessado em agosto 29, 2025, https://lab.rivas.ai/?p=2665
75. Friedman test and Nemenyi post-hoc test when comparing the time spent... -

ResearchGate, acessado em agosto 29, 2025, https://www.researchgate.net/figure/Friedman-test-and-Nemenyi-post-hoc-test-when-comparing-the-time-spent-by-the-different-AO_fig5_346555784
76. Nemenyi test - Wikipedia, acessado em agosto 29, 2025, https://en.wikipedia.org/wiki/Nemenyi_test
77. Common Language Effect Size - PS, acessado em agosto 29, 2025, https://peterstatistics.com/Terms/EffectSizes/CommonLanguageEffectSize.html
78. Wilcoxon Rank Sum Test and Vargha-Delaney Â12 Measure for the Traditional Metrics-based Models. - ResearchGate, acessado em agosto 29, 2025, https://www.researchgate.net/figure/Wilcoxon-Rank-Sum-Test-and-Vargha-Delaney-A12-Measure-for-the-Traditional-Metrics-based_tbl3_328210662
79. Average Vargha-Delaney effect sizes b A12 of each algorithm (in a row)... - ResearchGate, acessado em agosto 29, 2025, https://www.researchgate.net/figure/Average-Vargha-Delaney-effect-sizes-b-A12-of-each-algorithm-in-a-row-with-another_tbl3_349080081
80. Benchmarking optimization software with performance profiles. (Journal Article) | OSTI.GOV, acessado em agosto 29, 2025, https://www.osti.gov/biblio/943229
81. Benchmarking Optimization Software with Performance Profiles, acessado em agosto 29, 2025, https://www.researchgate.net/publication/1853672_Benchmarking_Optimization_Software_with_Performance_Profiles
82. Performance Profile Benchmarking Tool - Tangi Migot, acessado em agosto 29, 2025, https://tmigot.github.io/posts/2024/06/teaching/
83. Dolan, E.D. and Moré, J.J. (2002) Benchmarking Optimization Software with Performance Profiles. Mathematical Programming, 91, 201-213. - References, acessado em agosto 29, 2025, https://www.scirp.org/reference/referencespapers?referenceid=2663487
84. benchmarking derivative-free optimization algorithms - CiteSeerX, acessado em agosto 29, 2025, https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=90319b0b2a3c700015303fa2f30dcdc22f081ac1
85. Introduction to Performance Profile | Abel Soares Siqueira, acessado em agosto 29, 2025, https://abelsiqueira.com/blog/2017-05-09-introduction-to-performance-profile/
86. A note on performance profiles for benchmarking software - ePubs, acessado em agosto 29, 2025, https://epubs.stfc.ac.uk/manifestation/20477017/RAL-P-2015-004.pdf
87. A Note on Performance Profiles for Benchmarking Software | Request PDF - ResearchGate, acessado em agosto 29, 2025, https://www.researchgate.net/publication/310824952_A_Note_on_Performance_Profiles_for_Benchmarking_Software
88. JuliaSmoothOptimizers/BenchmarkProfiles.jl: Performance and data profiles - GitHub, acessado em agosto 29, 2025, https://github.com/JuliaSmoothOptimizers/BenchmarkProfiles.jl
89. Extending time-to-target plots to multiple instances and targets ... - UFF,

acessado em agosto 29, 2025, http://www.dcc.ic.uff.br/~celso/artigos/mtttplot-mic2017.pdf
90. tttplots-compare: a perl program to compare time-to-target plots or general runtime distributions of randomized algorithms | Request PDF - ResearchGate, acessado em agosto 29, 2025, https://www.researchgate.net/publication/271741508_tttplots-compare_a_perl_program_to_compare_time-to-target_plots_or_general_runtime_distributions_of_randomized_algorithms
91. tttplots: a perl program to create time-to-target plots - UFF, acessado em agosto 29, 2025, http://profs.ic.uff.br/~celso/artigos/tttplots.pdf
92. Best Practices for Comparing Optimization Algorithms - arXiv, acessado em agosto 29, 2025, https://arxiv.org/pdf/1709.08242
93. (PDF) COCO – COmparing Continuous Optimizers : The Documentation - ResearchGate, acessado em agosto 29, 2025, https://www.researchgate.net/publication/267853289_COCO_-_COmparing_Continuous_Optimizers_The_Documentation
94. COCO: Performance Assessment, acessado em agosto 29, 2025, https://numbbo.github.io/coco-doc/perf-assessment/
95. (PDF) COCO: Performance Assessment - ResearchGate, acessado em agosto 29, 2025, https://www.researchgate.net/publication/302951878_COCO_Performance_Assessment
96. Volume 13, Issue 4, acessado em agosto 29, 2025, https://evolution.sigevo.org/issues/HTML/sigevolution-13-4/index.htm
97. ecdf - Empirical cumulative distribution function - MATLAB - MathWorks, acessado em agosto 29, 2025, https://www.mathworks.com/help/stats/ecdf.html
98. ecdf — SciPy v1.16.1 Manual, acessado em agosto 29, 2025, https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.ecdf.html
99. Compute an ECDF for Censored Data - R, acessado em agosto 29, 2025, https://search.r-project.org/CRAN/refmans/NADA/html/cenfit.html
100. Compute an ECDF and Distribution Parameters for Censored Data — cfit • NADA2, acessado em agosto 29, 2025, https://swampthingecology.org/NADA2/reference/cfit.html
101. Using the Empirical Attainment Function for Analyzing Single-objective Black-box Optimization Algorithms - arXiv, acessado em agosto 29, 2025, https://arxiv.org/html/2404.02031v1
102. Empirical distribution function - Wikipedia, acessado em agosto 29, 2025, https://en.wikipedia.org/wiki/Empirical_distribution_function
103. Censoring Sensitivity Analysis for Benchmarking Survival Machine Learning Methods, acessado em agosto 29, 2025, https://www.mdpi.com/2413-4155/7/1/18
104. Benchmarking algorithms for gene regulatory network inference from single-cell transcriptomic data - PMC - PubMed Central, acessado em agosto 29, 2025, https://pmc.ncbi.nlm.nih.gov/articles/PMC7098173/
105. Censored Data, acessado em agosto 29, 2025, https://pdixon.stat.iastate.edu/stat505/Chapter%2011.pdf

106. NeurIPS 2024 Datasets Benchmarks 2024, acessado em agosto 29, 2025, https://neurips.cc/virtual/2024/events/datasets-benchmarks-2024
107. Benchmarking quantum optimization for the maximum-cut problem on a superconducting quantum computer | Phys. Rev. Applied - Physical Review Link Manager, acessado em agosto 29, 2025, https://link.aps.org/doi/10.1103/PhysRevApplied.23.014045
108. The five pillars of computational reproducibility: Bioinformatics and beyond - ResearchGate, acessado em agosto 29, 2025, https://www.researchgate.net/publication/371671830_The_five_pillars_of_computational_reproducibility_Bioinformatics_and_beyond
109. five pillars of computational reproducibility: bioinformatics and ..., acessado em agosto 29, 2025, https://academic.oup.com/bib/article/24/6/bbad375/7326135
110. The five pillars of computational reproducibility: bioinformatics and beyond - OUCI, acessado em agosto 29, 2025, https://ouci.dntb.gov.ua/en/works/7BKpE5Z4/
111. The five pillars of computational reproducibility. - ResearchGate, acessado em agosto 29, 2025, https://www.researchgate.net/figure/The-five-pillars-of-computational-reproducibility_fig1_374922562
112. The five pillars of computational reproducibility: bioinformatics and beyond - PubMed, acessado em agosto 29, 2025, https://pubmed.ncbi.nlm.nih.gov/37870287/
113. academic.oup.com, acessado em agosto 29, 2025, https://academic.oup.com/bib/article/24/6/bbad375/7326135#:~:text=These%20include%20(1)%20literate%20programming,easily%2C%20long%20into%20the%20future.
114. Git LFS and DVC: The Ultimate Guide to Managing Large Artifacts in ..., acessado em agosto 29, 2025, https://medium.com/@pablojusue/git-lfs-and-dvc-the-ultimate-guide-to-managing-large-artifacts-in-mlops-c1c926e6c5f4
115. Why Git LFS Is Not Good Practice for AI Model Weights and Why You Should Use DVC Instead (Demo with Azure Data Lake Storage 2) | by Neel Shah | Medium, acessado em agosto 29, 2025, https://medium.com/@neeldevenshah/why-git-lfs-is-not-good-practice-for-ai-model-weights-and-why-you-should-use-dvc-instead-demo-with-3903a7ae68f5
116. DVC vs Git vs Git LFS: ML Reproducibility - Censius, acessado em agosto 29, 2025, https://censius.ai/blogs/dvc-vs-git-and-git-lfs-in-machine-learning-reproducibility
117. What is the best way to do data version control for Machine Learning models - MATLAB Answers - MathWorks, acessado em agosto 29, 2025, https://www.mathworks.com/matlabcentral/answers/2067961-what-is-the-best-way-to-do-data-version-control-for-machine-learning-models
118. Difference between git-lfs and dvc - Stack Overflow, acessado em agosto 29, 2025,

https://stackoverflow.com/questions/58541260/difference-between-git-lfs-and-dvc
119. Conda vs Poetry in Python - GeeksforGeeks, acessado em agosto 29, 2025, https://www.geeksforgeeks.org/python/conda-vs-poetry-in-python/
120. Poetry vs Conda : r/Python - Reddit, acessado em agosto 29, 2025, https://www.reddit.com/r/Python/comments/t4g7xf/poetry_vs_conda/
121. Managing Python Dependencies with Poetry vs Conda & Pip - Exxact Corporation, acessado em agosto 29, 2025, https://www.exxactcorp.com/blog/Deep-Learning/managing-python-dependencies-with-poetry-vs-conda-pip
122. python - Does it make sense to use Conda + Poetry? - Stack Overflow, acessado em agosto 29, 2025, https://stackoverflow.com/questions/70851048/does-it-make-sense-to-use-conda-poetry
123. Conda and Poetry: A Harmonious Fusion | by Henrique Malta - Stackademic, acessado em agosto 29, 2025, https://blog.stackademic.com/conda-and-poetry-a-harmonious-fusion-8116895b6380
124. Using Make for Reproducible and Parallel Neuroimaging Workflow and Quality-Assurance, acessado em agosto 29, 2025, https://pmc.ncbi.nlm.nih.gov/articles/PMC4735413/
125. Reproducibility with Make - The Turing Way, acessado em agosto 29, 2025, https://book.the-turing-way.org/reproducible-research/make
126. Use Makefiles to Manage, Automate, and Reproduce Projects - Tilburg Science Hub, acessado em agosto 29, 2025, https://tilburgsciencehub.com/topics/automation/automation-tools/make-and-makefiles/what-are-makefiles/
127. Automation with make - Riffomonas, acessado em agosto 29, 2025, https://riffomonas.org/reproducible_research/make/
128. The Basic Reproducible Workflow Template · GitBook, acessado em agosto 29, 2025, http://www.practicereproducibleresearch.org/core-chapters/3-basic.html
129. Creating your first schema - JSON Schema, acessado em agosto 29, 2025, https://json-schema.org/learn/getting-started-step-by-step
130. Optimizing JSON Schema for Scalable Computer Vision Pipelines | Medium, acessado em agosto 29, 2025, https://medium.com/@noel.benji/designing-scalable-json-schemas-for-computer-vision-pipelines-dcddf4e7a9f4
131. Need to Verify Your JSON Schema? Here's a Few Ways to Do It ..., acessado em agosto 29, 2025, https://zuplo.com/blog/verify-json-schema
132. JSON Schema reference, acessado em agosto 29, 2025, https://json-schema.org/understanding-json-schema/reference
133. Introducing JSON Schemas for AI Data Integrity - DEV Community, acessado em agosto 29, 2025, https://dev.to/stephenc222/introducing-json-schemas-for-ai-data-integrity-611