Condução de Experimentos Computacionais com Métodos Heurísticos

Carine Rodrigues da Costa

Humberto Longo

Instituto de Informática
UFG, Goiânia–GO, Brasil
carinercrs@gmail.com
longo@inf.ufg.br

Resumo

A necessidade de resolver problemas de otimização em um limite razoável de tempo computacional faz com que o desenvolvimento de heurísticas seja uma grande área de pesquisa. Usualmente, heurísticas para problemas de otimização são desenvolvidas e avaliadas empiricamente. Mas ao descrever um experimento computacional e relatar os resultados, pode ficar evidente a dificuldade de reproduzir ou comparar os resultados obtidos com os de outros experimentos. Parte da origem dessas questões vem do fato de que não há padrão para o relato de experimentos na área de Computação. Portanto, este trabalho apresenta uma compilação com as recomendações de diversos autores sobre condução de experimentos computacionais com métodos heurísticos. Também é apresentado um *checklist* que sumariza essas recomendações, o qual pode ser usado para verificar os itens necessários para reprodução e comparação de experimentos.

Palavras-Chave: Otimização, Métodos Heurísticos, Experimentos Computacionais.
Área principal: MH – Metaheurísticas.

Abstract

The necessity of solving optimization problems in a reasonable limit of computational time makes the development of heuristics be a large research area. Usually, heuristics for optimization problems are developed and empirically evaluated, and describe a computational experiment and report the results, may become evident the difficulty to reproduce the experiment or compare the results with those of other experiments. Part of the origin of these issues comes from the fact that there is no standard for reporting experiments in Computer Science. Therefore, this paper presents a compilation with recommendations of several authors of conducting computational experiments with heuristic methods. Also provide a checklist that summarizes those recommendations.

Keywords: Optimization, Heuristic Methods, Computational Experiments.
Main area: MH – Metaheuristics.

1 Introdução

A necessidade de resolver problemas de otimização em um limite razoável de tempo faz com que o desenvolvimento de heurísticas seja uma grande área de pesquisa, bem como suas aplicações na resolução de problemas reais. Usualmente, heurísticas para problemas de otimização são desenvolvidas e avaliadas empiricamente, pela sua aplicação a um conjunto de instâncias específicas, comparando a qualidade de soluções e esforços computacionais. Mas, ao se apresentar uma nova heurística, esta deve ser descrita em detalhes, para que seja possível fazer uma análise sobre semelhanças e diferenças em relação a outras abordagens e permitir a comparação ou reprodução dos resultados obtidos com uma nova heurística.

Em geral, um experimento computacional com heurísticas consiste na resolução de uma série de instâncias do problema em questão. Assim, o pesquisador deve prover uma implementação computacional da heurística, selecionar as instâncias, escolher um ambiente computacional, escolher as medidas de desempenho, configurar os parâmetros da heurística e, então, relatar os resultados, geralmente o comportamento da heurística com as instâncias selecionadas. A condução de cada uma destas etapas pode ter um efeito substancial sobre os resultados e a relevância do experimento.

A descrição de um experimento computacional e o relato dos resultados obtidos pode tornar evidente a dificuldade de reprodução ou de comparação com outros resultados. Ao pensar sobre a melhor forma de conduzir um experimento, da definição dos objetivos ao relato dos resultados, surgem questões como: Quais os passos a serem seguidos na condução de experimentos computacionais com métodos heurísticos? O que realmente deve ser relatado em um experimento computacional utilizando heurísticas? Quais os requisitos mínimos para tornar um trabalho passível de comparação e reprodução? Parte da origem destas questões vem do fato que não há padrão para o relato de experimentos na área de Computação.

O trabalho fundamenta-se em tentar responder as questões que foram levantadas. Portanto, foi desenvolvida uma extensa investigação sobre condução de experimentos computacionais, com um levantamento de métodos empregados por diferentes pesquisadores, identificado um conjunto de recomendações que foram organizadas, pois estavam fragmentadas na literatura, para melhorar a condução de um experimento computacional utilizando heurísticas. Também foi elaborado um *checklist*, representando de forma sumarizada todos os itens vistos nesta investigação.

Dessa forma, este trabalho apresenta na Seção 2 as principais etapas na condução de experimentos com heurísticas, na Seção 3, um *checklist* que reúne os itens necessários para avaliação de relatos de experimentos computacionais com métodos heurísticos. Por fim, a Seção 4 apresenta as considerações finais.

2 Experimentos com Heurísticas

As principais etapas para uma boa condução de experimentos computacionais com heurísticas, segundo Crowder et al. (1979), Barr et al. (1995), McGeoch (1996), Johnson (2001), Rardin and Uzsoy (2001), Moret (2002), dentre outros, são: fazer a revisão da literatura; definir os objetivos do experimento; escolher medidas de desempenho e fatores a explorar; projetar e executar o experimento; analisar os dados; e relatar os resultados dos experimentos. Nas subseções 2.1 a 2.6 essas etapas são descritas e são listadas algumas das principais abordagens recomendadas.

2.1 Revisão da Literatura

A revisão da literatura ajuda a definir os objetivos da pesquisa, tornando possível obter as informações necessárias para elaboração de definições, suposições, limitações e hipóteses. Johnson (2001) afirma que um fator chave para publicar um artigo é contextualizá-lo em

relação ao estado da arte. Segundo McGeoch and Moret (1999) e Moret (2002), deve-se prover o contexto da pesquisa, ou seja, deve-se saber o que já foi estudado e desenvolvido tanto para o problema, quanto para os algoritmos. Um ponto importante é evitar desenvolver algo que já foi feito.

2.2 Objetivos do Experimento

Para Barr et al. (1995) e Moret (2002), um experimento deve ter um objetivo bem definido. A partir dele é que serão respondidas as questões, para as quais a experimentação é necessária. É nesta fase que o pesquisador lista as hipóteses a testar, os resultados a procurar e quais fatores explorar.

Moret (2002) cita uma lista de possíveis objetivos de pesquisa: **Testar e melhorar algoritmos para problemas difíceis** – entender como uma heurística trabalha para diminuir o tempo computacional ou delimitar a qualidade das aproximações obtidas; **Comparar algoritmos existentes e estruturas de dados para problemas** – fazer experimentos facilita a identificação de implementações boas ou ruins, e se a melhoria obtida pela teoria também é válida na prática. Novas conclusões podem ser inferidas para contribuir para um refinamento ou simplificação de um algoritmo; **Comprovar e Refinar Conjecturas** – testar conjecturas sobre uma série de casos pode, no mínimo, evitar fazer um trabalho que poderá ser desperdiçado futuramente e bons experimentos são uma fonte rica para novas conjecturas e teoremas; **Desenvolver instâncias de teste** – desenvolver algoritmos que gerem instâncias aleatórias, de forma que essas consigam abranger a maioria das características do problema em questão; **Desenvolver bibliotecas para algoritmos básicos e estruturas de dados** – deve-se implementar algoritmos que garantam que o tempo de execução seja eficiente e deve-se documentar os casos em que ele tem um desempenho bom ou ruim; e **Desenvolver ferramentas para facilitar o projeto e análise de algoritmos** – nesta categoria se enquadram ferramentas gráficas e estatísticas para analisar experimentos, podendo conter ferramentas de animação para visualizar o progresso de um experimento.

2.3 Medidas de Desempenho e Fatores a Explorar

Podem ser encontradas na literatura (Ahuja et al. (1993), Ahuja and Orlin (1992), Barr et al. (1995), McGeoch (1992), Moret (2002), Rardin and Uzsoy (2001)) várias medidas de desempenho para avaliação de métodos heurísticos. Essas medidas de desempenho podem ser divididas em três áreas: **qualidade da solução**, **esforço computacional** e **robustez**.

Uma medida que pode mostrar uma estimativa sobre a qualidade da solução é a acurácia, que é a diferença entre o valor encontrado e o valor de referência. O valor de referência pode ser uma solução encontrada por meio de métodos exatos, ou pode ser o melhor valor encontrado por alguma heurística (Lilja (2004)). Na realidade é um grande desafio avaliar a qualidade das soluções encontradas por heurísticas, pois geralmente os problemas resolvidos com heurísticas são NP-Difíceis, e muitas vezes para determinados problemas, métodos exatos não produzem soluções de qualidade em tempo viável. Assim, a utilização da qualidade da solução como medida de desempenho acaba empregando métodos como: cálculo da solução exata para pequenas instâncias; uso de limites inferiores ou superiores; construção de instâncias a partir de valores ótimos conhecidos; estimativa estatística de valores ótimos conhecidos; e comparação dos melhores valores encontrados (Rardin and Uzsoy (2001)).

O esforço computacional, em especial a velocidade de computação, também é um fator chave. De acordo com Barr et al. (1995), várias partes do processo podem ser cronometradas, como: tempo da melhor solução encontrada, tempo médio total de execução ou tempo por fase. Segundo Johnson (2001), mesmo que o objetivo principal não seja a análise dos tempos de execução, eles devem ser relatados, pois pode ser de interesse do leitor. Por exemplo, se as principais operações são cálculos combinatórios e operações algorítmicas, pode-se querer

fazer uma correlação dessas operações com o tempo de execução da heurística.

Uma heurística que obtém uma solução excelente para apenas uma instância do problema não é robusta e também não é interessante. Em geral, a robustez é baseada na capacidade da heurística de encontrar soluções para uma grande variedade de instâncias e é mostrada com as medidas de variância (Barr et al. (1995), Crowder et al. (1979), Greenberg (1990)).

Além dos fatores qualidade da solução, esforços computacionais e robustez, Crowder et al. (1979, 1978) apontam outros indicadores de desempenho. Os principais são: *tradeoffs* – uma comparação feita entre dois fatores, onde é verificado qual fator perde ou ganha na execução de um experimento (tenta-se achar um equilíbrio para os dois fatores, com o objetivo de ter um bom desempenho nos testes); a precisão numérica – uma medida de capacidade do algoritmo de computar corretamente a resposta diante da instabilidade numérica; a quantidade de iterações – a quantidade de passos que o algoritmo necessita para resolver o problema (independe do computador usado); quantidade de chamadas de uma determinada função – pode ser chamada da própria função objetivo ou de outra função que tenha relevância no teste; operações matemáticas – a quantidade de vezes que uma operação básica é necessária durante a execução do algoritmo. Além disso, medidas independentes de ambiente computacional como, por exemplo, a altura de uma árvore de busca, a quantidade de nós ou o tempo de execução por nó, também podem facilitar a comparação (Hooker (1995)).

2.4 Projeto e Execução do Experimento

O modelo experimental influencia toda a condução do experimento. Os objetivos levantados no início do experimento devem ser claros, para que ao final, seja possível obter as conclusões esperadas. Um bom experimento deve alcançar as metas experimentais, demonstrar claramente o desempenho dos testes, ter justificativas lógicas, gerar boas conclusões e ser passível de reprodução. Todas estas características têm um valor importante nos testes dos métodos heurísticos. Além de escolher um modelo experimental, também são feitas nesta fase a seleção ou geração do conjunto de instâncias de teste, são executados os testes e feitos ajustes nos parâmetros. As Subseções 2.4.1 à 2.4.4 detalham essas etapas.

2.4.1 Modelo Experimental

O conjunto de parâmetros considerados para analisar uma heurística para solucionar um dado problema (modelo experimental), deve ser definido cuidadosamente, para que se possa fazer inferências sobre o desempenho da mesma. Produzir um modelo experimental de confiança envolve: identificar as variáveis que podem influenciar no desempenho; decidir as medidas apropriadas de desempenho e avaliar a variância destas medidas, selecionando um conjunto apropriado de instâncias de teste para poder responder às questões que são levantadas nos objetivos do experimento (Crowder et al. (1979, 1978)). Rardin and Uzsoy (2001) citam quatro tipos de modelos experimentais: Básico – instâncias $\times$ algoritmos, Modelo Experimental Refinado – também chamado de Planejamento Estatístico de Experimentos, Blocagem de Instâncias e Balanceamento de Qualidade e Tempo.

O modelo Básico corresponde a um planejamento experimental simples, em que os dados são armazenados em uma tabela, onde as linhas correspondem às instâncias do problema e as colunas aos algoritmos testados. Cada célula da tabela equivale a um tempo de execução de um algoritmo sobre uma instância. Os algoritmos podem ser diferentes, mas geralmente são variações de uma mesma ideia. As instâncias podem ser completamente diferentes e independentes, mas geralmente são organizadas por características similares, como o tamanho.

O Modelo Experimental Refinado inicia com um conjunto de questões sobre as heurísticas em estudo que precisam ser respondidas. Tais questões, em geral dizem respeito a como as diferentes características do problema (tais como tamanho do problema, quantidade e natureza das restrições) e parâmetros dos algoritmos (critério de parada, busca na vizinhança e seleção

de movimentos) afetam o desempenho das heurísticas testadas. Definidas as características (fatores), que podem possuir diferentes valores (níveis), ou seja, variáveis cujos efeitos são de interesse da investigação, o modelo consiste em executar todas as combinações entre os níveis dos fatores e, ao final, avaliar os resultados para ver o que pode ser concluído.

No processo de obtenção de conclusões válidas e objetivas, para minimizar a quantidade de testes e maximizar a informação adquirida, pode-se estruturar os experimentos de forma a garantir que os dados coletados sejam analisados por modelos estatísticos. Existem métodos, tais como a Análise de Variância, o Fatorial Completo, ou Quadrado Latino, que podem ser muito utilizados para análise estatística de experimentos (Montgomery (2009)). Quando os resultados de um experimento variam de acordo com as condições de teste (parâmetros de configuração, ambiente computacional, problemas resolvidos), a metodologia estatística é a única abordagem objetiva de análise. Por isso, o projeto de um experimento e análise estatística dos dados são inter-relacionados (Rardin and Uzsoy (2001)).

O planejamento estatístico de experimentos define que os fatores não controláveis devem ser definidos de forma aleatória. A Blocagem de Instâncias usa técnicas de randomização. Assim, a aplicação desse princípio deve ser rigorosa, pois instâncias aleatórias e diferentes devem ser escolhidas para cada algoritmo. Em geral, a blocagem de instâncias é feita testando todos os algoritmos com o mesmo conjunto de instâncias de teste. Lin and Rardin (1979) demonstram que diferenças entre algoritmos são mais fáceis de serem encontradas estatisticamente se os mesmos conjuntos de instâncias forem testados por todos os algoritmos.

Uma das principais características em experimentos computacionais é a necessidade de descrever os *tradeoffs* entre tempo para obtenção de uma solução e quão perto ela é de uma solução ótima. O princípio da abordagem Balanceamento de Qualidade e Tempo é que todos os algoritmos consumam a mesma quantidade de recursos computacionais. Essa é, muitas vezes, o modo mais eficaz para comparar algoritmos heurísticos bastante diferentes.

Embora modelos experimentais sejam muito empregados em áreas como Física, Engenharia ou Medicina, padrões para testes empíricos em Computação têm sido menos rigorosos e por isso há uma ampla gama de métodos aceitos para a análise de dados, os quais não utilizam todo o rigor estatístico. No entanto, os resultados experimentais devem ser avaliados para o entendimento do problema e dos algoritmos em estudo, ao invés de mostrar simplesmente que um algoritmo é melhor que outro. Portanto, as técnicas de avaliação de algoritmos utilizando conceitos estatísticos são uma opção para delinear um experimento computacional.

2.4.2 Seleção do Conjunto de Instâncias de Teste

Uma das maiores tarefas na condução de experimentos computacionais é construir ou selecionar as instâncias de teste, pois não importa o quão bem feita é a estrutura do projeto experimental, se não houver disponibilidade de dados suficientes, reais e com variedade que abranja todas as características do problema em questão. São importantes para verificar a efetividade de uma dada heurística, e com isso, obter inferências que serão usadas em situações práticas. Rardin and Uzsoy (2001) citam quatro métodos para obter um conjunto de instâncias de teste: uso de dados de situações reais (instâncias reais); variações de instâncias reais; bibliotecas públicas de referência e instâncias geradas aleatoriamente.

Em geral, os melhores conjuntos de instâncias de testes são aqueles com instâncias reais. Entretanto, pode ser difícil obtê-las, pois empresas ou instituições podem não tornar os dados públicos, ou só aceitam, caso informações como nome e outros dados importantes sejam omitidos. A coleta de dados reais também pode levar um tempo considerável. Uma alternativa, pouco usada, mas que pode tornar uma instância mais complexa e com isso ser mais trabalhoso encontrar uma solução, é substituir valores reais por valores aleatórios. A macro-estrutura é preservada, mas detalhes são modificados aleatoriamente para produzir novas instâncias. As bibliotecas públicas de referência são muito utilizadas, e existem muitos

repositórios de instâncias de teste (vide OR-Library (2010), Netlib (2010), TSPLib (2010), Burkard et al. (1996)) para muitas classes de problemas clássicos.

Pesquisas feitas no início do ciclo de vida de um problema muitas vezes apresentam poucas instâncias. Como a pesquisa continua, os conjuntos de instâncias de teste utilizados pelos pioneiros das pesquisas tendem a se tornar coleções de referências clássicas, utilizadas por todos os pesquisadores que trabalham sobre o mesmo problema. Em geral, os conjuntos de instâncias de referência são formados a partir da contribuição de vários pesquisadores. Todavia, esses conjuntos podem apresentar algumas falhas. Por exemplo, muitas vezes não há como saber se a instância que foi adicionada ao conjunto de referência é real ou aleatória.

Outra questão é que algumas das instâncias encontradas em bibliotecas de referência não têm o objetivo de representar aplicações reais. Podem servir para avaliar um algoritmo em todos os seus passos, ou simplesmente mostrar um comportamento estranho. Também pode ocorrer que as instâncias escolhidas sejam as que resultaram em bons resultados para os algoritmos desenvolvidos. Isto pode resultar em instâncias com padrões ocultos, ou seja, algum pesquisador construiu instâncias em que foram obtidas boas soluções para seus algoritmos, mas não se sabe se estas instâncias terão bons resultados com outros algoritmos.

Por fim, as instâncias geradas aleatoriamente, de forma totalmente artificial e com propriedades controladas por parâmetros gerais, é a maneira mais fácil e rápida de se obter um conjunto de instâncias de teste. As características da instância são controladas por quem a gerou, o que permite a produção de diversas populações de instâncias, englobando características que muitas vezes não são encontradas em instâncias reais. Essa abordagem permite que sejam geradas instâncias em que uma solução ótima é conhecida, o que facilita a avaliação do desempenho das heurísticas. Entretanto, estas instâncias podem gerar conclusões totalmente distorcidas em relação ao mundo real e muitas vezes criticadas por retratarem situações irreais e serem mais difíceis de serem resolvidas que os problemas reais. De qualquer forma, se as instâncias são geradas pelo pesquisador, então o processo de geração deve ser claramente descrito, para ser utilizado por outros pesquisadores (Barr et al. (1995), Johnson (2001)).

2.4.3 Execução do Experimento

A quantidade de testes necessários varia de acordo com os objetivos delineados inicialmente no projeto experimental. Johnson (2001) afirma que um erro que não se deve cometer é realizar apenas um teste com cada instância, pois pode-se chegar a conclusões erradas e até mesmo tornar o experimento irreprodutível. Por outro lado, deve-se atentar para o fato de que métodos que possuem componentes com aleatoriedade podem dar resultados diferentes ao fazer vários testes com uma única instância. Para reduzir a variância dos resultados, deve-se testar várias instâncias, e se são utilizados geradores aleatórios, o código deve ser testado sobre cada instância com sementes de inicialização diferentes.

Outro problema citado por Johnson (2001) é a utilização do melhor resultado encontrado como critério de avaliação, principalmente quando são estudados algoritmos aleatórios. Geralmente são apresentadas tabelas que contêm o melhor resultado encontrado ou a média dos resultados encontrados. Mas existem pelo menos dois problemas em relação a essa abordagem: a melhor solução encontrada é somente uma amostra da distribuição de soluções existentes e menos provável de ser reprodutível do que a média; e os tempos de execução relatados são geralmente uma única execução do algoritmo, ao invés de todas as execuções que foram feitas, e representam apenas o tempo para encontrar a melhor solução. Se a quantidade de execuções é descrita, o simples procedimento de multiplicar o tempo de execução pela quantidade de execuções pode superestimar ou subestimar o tempo necessário, embora alguns passos necessitem ser realizados apenas uma vez, como a leitura da instância e a criação das estruturas de dados.

2.4.4 Ajustes de Parâmetros

Ao configurar parâmetros (fatores) em metaheurísticas como *Simulated Annealing*, Busca Tabu ou Algoritmos Genéticos, pode-se: fixar os parâmetros para todas as instâncias ou configurar os parâmetros para cada instância testada. Quando os parâmetros são fixos não há problema em reproduzir o experimento, mas deve-se descrever as configurações utilizadas. Mas se as configurações variam com a instância, deve-se explicar o porquê da utilização destas. Se determinadas configurações não estão bem especificadas, pode significar que o algoritmo está mal especificado ou mal implementado. Devem ser documentados e justificados os parâmetros associados com alguma regra de parada e os valores de parâmetros, ou regras usadas para resolver cada instância, quando esses diferem nas várias instâncias testadas. Além disso, o tempo necessário para a configuração dos parâmetros não deve ser incluído no tempo total de execução, pois pode parecer que o algoritmo leva mais tempo para executar do que na realidade (Barr et al. (1995), Rardin and Uzsoy (2001)).

2.5 Análise de Dados e Relato do Experimento

Em geral, após a execução de um experimento computacional, deve-se analisar uma grande quantidade de dados, obtidos pelos testes de diversos algoritmos sobre um conjunto de instâncias de teste. Essa análise consiste na avaliação dos dados obtidos com a aplicação de técnicas estatísticas e não estatísticas, visando os objetivos definidos no início do experimento (Barr et al. (1995)). Algumas das principais medidas de análise são o valor da solução encontrada, médias encontradas pela execução de vários testes, diferenças entre valor encontrado e solução ótima, valores máximo e mínimo encontrados, desvio padrão, erro amostral e tamanho da amostra utilizada (Crowder et al. (1979)). Podem-se considerar os *tradeoffs* chaves, como a qualidade da solução em relação ao tempo ou velocidade em relação à robustez.

A média e o desvio padrão da qualidade de soluções obtidas com cada heurística, dado um conjunto fixo de instâncias de teste, podem ser calculadas. Como a população de instâncias é fixa, não há erros nestas estatísticas. Pode haver algum erro nas medidas de tempo de execução, que podem variar de acordo com o ambiente computacional, mas os valores das soluções permaneceriam os mesmos, caso fossem testados novamente sobre o mesmo conjunto de instâncias. Assim, como não há incerteza na amostragem, não são necessários métodos estatísticos para análise destes resultados Rardin and Uzsoy (2001).

A incerteza existe nos experimentos computacionais quando os resultados podem ser vistos como uma amostra aleatória, grande e com uma população essencialmente infinita. Já que não há como testar toda a população, é possível beneficiar-se dos métodos estatísticos, para se ter uma ideia de quão confiáveis são os dados. A aleatoriedade está presente nos experimentos computacionais quando instâncias são construídas com geradores aleatórios, ou quando algoritmos tomam decisões aleatórias em sua busca. Metaheurísticas como Algoritmos Genéticos e GRASP, por exemplo, tomam decisões aleatórias em cada etapa, e em cada decisão é utilizada apenas uma amostra da população de resultados para cada combinação de instância e algoritmo.

Quando resultados computacionais são obtidos de amostras aleatórias de uma população grande, qualquer análise deve atentar para a possibilidade de que efeitos aparentes nos resultados não sejam simples acidentes estatísticos. A técnica de significância estatística, uma probabilidade de erro na aceitação de um resultado observado como válido, representa um índice decrescente da confiabilidade de um resultado. Portanto, quanto mais alto o nível de significância, menos se pode acreditar que a relação observada entre as variáveis da amostra representa um indicador confiável da relação entre as respectivas variáveis na população. Alguns trabalhos com uso de métodos estatísticos, tais como os de Barr et al. (1995), Hooker (1995) e McGeoch (1996), defendem o uso de testes formais de significância estatística como uma maneira de introduzir mais precisão científica nas investigações empíricas de algoritmos.

Mesmo com sua importância, é fundamental ter em mente as limitações dos testes formais de significância estatística. Todo teste é construído sob uma série de suposições que pode ser questionada quando se trata de experimentos com heurísticas. Certamente, estatísticas formais não podem render conclusões muito úteis, a menos que sejam feitas com cuidado e corretamente. Pode surgir a dificuldade em trabalhar com os métodos estatísticos e dados computacionais, pois é necessário experiência para organizar os dados e selecionar os métodos corretos a serem utilizados para análise.

A análise de variância de um fator é uma técnica estatística que pode ajudar na análise dos resultados. A ideia básica dessa análise é assumir que toda variação não aleatória nas observações experimentais é devida às diferenças de desempenho médio nos níveis alternativos dos fatores experimentais (Montgomery (2009)). Em geral, a aplicação desse método permite inferir diversas conclusões a respeito dos dados. A técnica pode ser aplicada para ajudar na fase de análise da pesquisa, a deduzir quais são os principais fatores experimentais com maior significância estatística, que podem ser os algoritmos ou o tamanho das instâncias, por exemplo. As suas principais fontes de informação são as interações que mostram o impacto de como um fator depende do nível de outro. Algo interessante é mostrar as diferenças percentuais entre as médias encontradas. Técnicas de estimativas são consideradas promissoras, pois os esforços computacionais para calcular estimativas de valores ótimos e limites de confiança são relativamente pequenos, e as técnicas oferecem a esperança de obter informação confiável sobre valores de soluções ótimas que são independentes do problema de domínio.

2.6 Relato dos Resultados dos Experimentos

No âmbito de conseguir demonstrar que uma pesquisa apresenta alguma contribuição, o leitor deve ser convencido disso através de um relatório feito com qualidade. Para Johnson (2001) descrever de forma bem justificada o comportamento de um algoritmo é, sem dúvida, um dos principais objetivos de experimentos com algoritmos. Um bom relatório deve ser claro o suficiente para permitir a reprodução dos experimentos e a comparação dos resultados, para tornar possível a continuação da pesquisa desenvolvida. Isto requer o relato detalhado do modelo experimental utilizado, o relato dos testes, detalhes sobre os algoritmos desenvolvidos e implementação, descritos em detalhes suficientes que permitam a replicação (Crowder et al. (1979), Barr et al. (1995), Johnson (2001), Rardin and Uzsoy (2001)). A descrição do modelo experimental, descrição dos algoritmos, detalhes de implementação e conclusões são úteis para que outros pesquisadores possam ao menos fazer os experimentos de forma similar e chegar às mesmas conclusões. Detalhes que devem ser citados encontram-se na Tabela 1.

Com uma análise de dados bem feita, é possível justificar as conclusões a partir dos dados que foram apresentados, bem como direcionar pesquisas futuras e possíveis melhoras no algoritmo. Uma falha comumente encontrada em vários textos é a apresentação de dados sem interpretação. Não é suficiente realizar os testes, colocar os resultados em tabelas e deixar que o leitor tire suas próprias conclusões (Johnson (2001), McGeoch and Moret (1999)). No mínimo devem ser relatados os padrões encontrados nos dados. Afinal, se as questões forem bem delineadas, o experimento dará alguma resposta. Entretanto, Rardin and Uzsoy (2001), McGeoch and Moret (1999) afirmam que não há razão para que um texto inclua todos os detalhes para reproduzir o estudo. Contudo, estas informações devem estar disponíveis em um documento de trabalho, memorando ou relatório técnico, que contenha todos os detalhes necessários para a reprodução dos resultados.

McGeoch (1996), Johnson (2001) listam alguns padrões que, se utilizados, podem ser irreproduzíveis: relatar somente o valor da solução, pode tornar o experimento irreproduzível em um sentido limitado, mas não no sentido mais amplo, em que pode-se realizar experiências em casos semelhantes para comparar se os resultados são semelhantes; relatar somente a

porcentagem sobre a melhor solução calculada não representa muito, devido ao fato de que se forem calculadas várias soluções, não há como afirmar que elas são as melhores, por isso, deve-se sempre fornecer a instância ou o valor encontrado; relatar a porcentagem sobre uma estimativa da solução ótima esperada, para instâncias geradas aleatoriamente, pode ter resultados significativos se a estimativa é de fato consistente e perto do ideal esperado e se os valores ótimos encontrados têm variância relativamente baixa; relatar a porcentagem excedente do limite inferior, será reprodutível se o limite inferior pode ser calculado facilmente ou possível de fazer um cálculo aproximado.

Uma compreensão melhor dos dados obtidos pode vir da análise do conjunto completo dos dados e não apenas das estatísticas resumidas. Gráficos oferecem um meio de visualizar todos os dados, mas esses sem tabelas não possibilitam a compreensão dos dados, pois as figuras permitem dar uma ideia geral, contudo não apresentam detalhes dos resultados. Enfim, um bom texto deve conter tabelas e gráficos. Na construção de um gráfico deve-se entender bem o conjunto de dados, para que se possa gerar representações que revelem características marcantes nos dados. Quando um grande conjunto de dados é analisado, é útil que sejam organizados e resumidos numa tabela de distribuição de frequência, que lista os valores dos dados com os suas respectivas frequências. Esta tabela auxilia o entendimento da natureza da distribuição dos dados, que pode ser em forma de sino, uniforme ou assimétrica. Existem muitas técnicas para exibir resultados, que dependem dos objetivos que se quer alcançar. Para um olhar mais crítico sobre esta questão é aconselhável procurar trabalhos de referência, tais como o de Cleveland (1993) ou Triola (1999).

Johnson (2001) menciona uma preocupação quanto aos resultados anormais, ou seja, que diferem do valor esperado ou que são inconsistentes com as conclusões desejadas. Estes resultados não devem ser omitidos quando são encontrados. Não deve-se esquecer que a pior anomalia é aquela que o pesquisador não percebe. Isso pode deixar o leitor em dúvida, porque leva-o a pensar que poderia ser ou um simples erro de digitação ou um resultado anormal.

3 Avaliação de Relato de Experimento Computacional

Com o objetivo de facilitar a análise de relatos de experimentos computacionais, foi desenvolvido, a partir das observações de Crowder et al. (1979), o *checklist* que se encontra na Tabela 1. O *checklist* sumariza todas as recomendações descritas na Seção 2, com a divisão dos itens de forma clara e objetiva. O seu objetivo é servir como um guia que auxilie tanto na condução de um experimento quanto no relato dos resultados do mesmo. O *checklist* foi dividido em seis partes, de A a F, organizadas segundo as recomendações descritas nas Seções 2.1 à 2.6. Cada item resume uma recomendação e referencia a seção do texto que a detalha. A última coluna, chamada Peso, é dedicada a atribuir valores de acordo com a análise a ser feita. Dessa forma, além de ser um guia para elaboração de relatos de experimentos, pode ser utilizado, por exemplo, por revisores para avaliação de artigos em conferências. Se for utilizado para avaliação de artigos, pode-se atribuir pesos diferentes aos vários itens, adequando a avaliação a critérios previamente estabelecidos que os artigos devem atender. Por exemplo, se o modelo experimental é considerado mais importante que dados sobre implementação, dá-se um peso maior aos itens contidos no modelo experimental. Outra opção é desconsiderar os pesos e somente verificar se o texto segue ou não determinada recomendação.

Tabela 1: *Checklist* para Relato de Experimentos Computacionais.

A - Revisão da Literatura	
Itens Recomendados	**Peso**
1) O problema é novo? (Seção 2.1)	

2) O algoritmo proposto é novo? (Seção 2.1)	
3) O algoritmo já foi implementado? (Seção 2.1)	
4) O algoritmo já foi estudado para o problema em questão? (Seção 2.1)	
5) Fala sobre modelagens existentes? (Seção 2.1)	
6) Fala sobre métodos já desenvolvidos para o problema? (Seção 2.1)	
7) Definição clara do problema. (Seção 2.1)	

B - Modelo Experimental		
Itens Recomendados		**Peso**
1) Definição clara dos objetivos do experimento.	a) Compara a abordagem com técnicas já implementadas? (Seção 2.2)	
	b) Testa e melhora algoritmos para problemas difíceis? (Seção 2.2)	
	c) Compara algoritmos existentes e estruturas de dados? (Seção 2.2)	
	d) Comprova e refina conjecturas? (Seção 2.2)	
	e) Propõe biblioteca de algoritmos básicos e estruturas de dados? (Seção 2.2)	
	f) Propõe ferramentas para projeto e análise de algoritmos? (Seção 2.2)	
2) Modelo experimental	a) Básico (instâncias × algoritmos). (Seção 2.4)	
	b) Estatístico. (Seção 2.4)	
3) Descrição da execução do experimento	a) Quantidade de testes repetidos feitos por instância. (Seção 2.4)	
	b) Quantidade de instâncias testadas. (Seção 2.6)	
	c) Quantidade de sementes utilizadas. (Seção 2.6)	
	d) Quantidade máxima de iterações. (Seção 2.6)	
	e) Critério de parada utilizado. (Seção 2.6)	
4) Descrição das instâncias.	a) Instâncias reais ou aleatórias? (Seção 2.4)	
	b) Instâncias de referência? (Seção 2.4)	
5) Novas instâncias	a) Descrição do gerador de instâncias? (Seção 2.4)	

C - Apresentação dos Algoritmos		
Itens Recomendados		**Peso**
1) Descrição completa do algoritmo (Seção 2.6)		
2) Classe de problema que o algoritmo resolve.	a) Qual tipo de instâncias o algoritmo encontra soluções? (Seção 2.6	
	b) O algoritmo encontra soluções para instâncias de até que tamanho? (Seção 2.6)	
3) Descrição da técnica de estratégia inicial. (Seção 2.6)		
4) Uso de diferentes critérios de inicialização. (Seção 2.6)		
5) Uso de diferentes critérios de término. (Seção 2.6)		
6) Dados dos parâmetros da heurística (Ex: tamanho da lista tabu). (Seção 2.6)		
7) Uso de diferentes valores nos parâmetros do algoritmo. (Seção 2.6)		
8) Análise de complexidade do algoritmo. (Seção 2.6)		
9) Análise da quantidade de trabalho por iteração. (Seção 2.6)		

D - Implementação	
Itens Recomendados	**Peso**
1) Linguagem de programação. (Seção 2.6)	
2) Descrição dos dados de entrada. (Seção 2.6)	
3) Descrição das configurações. (Seção 2.6)	
4) Descrição de técnicas de pré-processamento. (Seção 2.6)	
5) Armazenamento dos requisitos e estruturas de dados. (Seção 2.6)	
6) Compilador. (Seção 2.6)	
7) Opções do compilador. (Seção 2.6)	
8) Sistema Operacional. (Seção 2.6)	
9) Hardware (modelo do computador, processador e memória). (Seção 2.6)	
10) Se o código está disponível. (Seção 2.6)	
11) Instruções para uso. (Seção 2.6)	

E - Relato e Análise dos Resultados		
Itens Recomendados		**Peso**
1) Medidas de desempenho	a) Qualidade da solução - acurácia de obtenção. (Seção 2.3)	
	b) Esforço computacional - Tempo da melhor solução encontrada. (Seção 2.3)	
	c) Esforço computacional - tempo médio total de execução. (Seção 2.3)	
	d) Esforço computacional - tempo por fase (se existirem fases). (Seção 2.3)	

	e) Robustez. (Seção 2.3)	
	f) Precisão numérica. (Seção 2.3)	
	g) Quantidade de iterações. (Seção 2.3)	
	h) Quantidade de chamadas de uma determinada função. (Seção 2.3)	
	i) Operações matemáticas. (Seção 2.3)	
2) Justificativa das medidas utilizadas. (Seção 2.5)		
3) Medidas de análise	a) Valor da solução encontrada. (Seção 2.5)	
	b) Médias. (Seção 2.5)	
	c) Totais. (Seção 2.5)	
	d) Diferenças. (Seção 2.5)	
	e) Valor mínimo encontrado. (Seção 2.5)	
	f) Valor máximo encontrado. (Seção 2.5)	
	g) Desvio padrão. (Seção 2.5)	
	h) Erro amostral. (Seção 2.5)	
	i) Tamanho da amostra utilizada. (Seção 2.5)	
4) Análise estatística	a) Intervalo de confiança. (Seção 2.5)	
	b) Nível de confiança. (Seção 2.5)	
	c) Margem de erro. (Seção 2.5)	
	d) Tamanho da amostra utilizada. (Seção 2.5)	
5) Uso de gráficos legíveis. (Seção 2.6)		
6) Uso de tabelas legíveis. (Seção 2.6)		

F - Conclusões	
Itens Recomendados	**Peso**
1) Justificar as conclusões a partir dos dados apresentados. (Seção 2.6)	
2) Identificação das instâncias que foram resolvidas com êxito. (Seção 2.6)	
3) Identificação das instâncias que não foram resolvidas. (Seção 2.6)	
4) Possíveis melhoras no algoritmo. (Seção 2.6)	
5) Direções nas pesquisas futuras. (Seção 2.6)	

4 Considerações Finais

As recomendações sobre condução de experimentos são, de um modo geral, bastante conhecidas e os estudos pioneiros datam desde 1979, como o trabalho de Crowder et al. (1979). Contudo, tais estudos não definem métricas nem diretrizes exatas, o que existe são apenas recomendações de boas práticas. Assim, este trabalho descreve uma extensa investigação sobre condução de experimentos computacionais, identificando e organizando um conjunto de recomendações fragmentadas na literatura e propõe um *checklist* que sumariza essas recomendações. Dado que os resultados de estudos experimentais na computação podem influenciar aplicações do mundo real, é de suma importância saber prepará-los, realizá-los e relatá-los com o máximo de cuidado. Apesar disso, pode-se identificar em diversos trabalhos muitas falhas no relato de experimentos que poderiam ser evitadas com o uso de boas práticas em sua condução. Além disso, é sempre desejável, para melhor entendimento dos experimentos, que o nível de qualidade dos artigos e relatórios seja o melhor possível.

Referências

Ahuja, R. K., Magnanti, T. L., and Orlin, J. B. (1993). *Network Flows: theory, algorithms, and applications*. Prentice-Hall.

Ahuja, R. K. and Orlin, J. B. (1992). Use of representative operation counts in computational testings of algorithms. Working papers 3459-92, Massachusetts Institute of Technology (MIT), Sloan School of Management.

Barr, R. S., Golden, B. L., Kelly, J. P., Resende, M. G. C., and Stewart, W. R. (1995). Designing and Reporting on Computational Experiments with Heuristic Methods. *Journal of Heuristics*, 1:9–32.

Burkard, R. E., Karisch, S. E., and Rendl, F. (1996). QAPLIB - A Quadratic Assignment Problem Library. http://www.opt.math.tugraz.at/qaplib/, último acesso em Fevereiro de 2011.

Cleveland, W. S. (1993). *Visualizing data*. AT & Bell Laboratories.

Crowder, H., Dembo, R. S., and Mulvey, J. M. (1979). On reporting computational experiments with mathematical software. *ACM Transactions on Mathematical Software*, 5(2):193–203.

Crowder, H. P., Dembo, R. S., and Mulvey, J. M. (1978). Reporting computational experiments in mathematical programming. *Mathematical Programming*, 15(1):316–329.

Golden, B. L., Assad, A. A., Wasil, E. A., and Baker, E. (1986). Experimentation in Optimization. *European Journal of Operational Research*, 27(1):1–16.

Golden, B. L., Stewart, W. R. I. L. E. L., Lenstra, J. K., Kan, A. H. G. R., and Shmoys, D. B. (1985). *Empirical Evaluation of Heuristics*, chapter 7, pages 207–250. John Wiley e Sons Ltd., New York.

Greenberg, H. J. (1990). Computational Testing: Why, How and How Much. *INFORMS Journal on Computing*, 2(1):94–97.

Hooker, J. (1994). Needed: An Empirical Science Of Algorithms. *Operations Research*, 42:201–212.

Hooker, J. (1995). Testing Heuristics: We Have It All Wrong. *Journal of Heuristics*, 1:33–42.

Jakcson, R. H. F., Boggs, P. T., Nash, S. G., and Powell, S. (1990). Guidelines for Reporting Results of Computational Experiments. Report of the Ad Hoc Committee. *Mathematical Programming*, 49(1):413–425.

Johnson, D. S. (2001). A Theoretician's Guide to the Experimental Analysis of Algorithms. In *Dagstuhl Seminar on Experimental Algorithmics*.

Lee, C.-Y., Bard, J., Pinedo, M., and Wilhel, W. E. (1993). Guidelines for Reporting Computational Results in IEEE Transactions. *IEEE Transactions*, 25(6):121–123.

Lilja, D. J. (2004). *Measuring Computer Performance, A Practitioner's Guide*. Cambridge University Press, 1 edition.

Lin, B. W. and Rardin, R. L. (1979). Controlled Experimental Design for Statistical Comparison of Integer Programming Algorithms. *MANAGEMENT SCIENCE*, 25(12):1258–1271.

McGeoch, C. C. (1992). Analyzing Algorithms by Simulation: Variance Reduction Techniques and Simulation Speedups. *ACM Computing Surveys*, 24(2):195–212.

McGeoch, C. C. (1996). Toward an Experimental Method for Algorithm Simulation. *INFORMS Journal on Computing*, 8(1):1–15.

McGeoch, C. C. and Moret, B. M. E. (1999). How to Present a Paper on Experimental Work with Algorithms. *SIGACT News*, 30(4):85–90.

Montgomery, D. C. (2009). *Design and Analysis of Experiments*. John Wiley & Sons, Inc., 7 edition.

Moret, B. M. E. (2002). Towards a discipline of experimental algorithmics. In Goldwasser, M. H., Johnson, D. S., and McGeoch, C. C., editors, *Data Structures, Near Neighbor Searches, and Methodology: Fifth and Sixth DIMACS Implementation Challenges*, volume 59 of *DIMACS Monographs*, pages 197–213. AMS Press.

Netlib (2010). Netlib Repository at UTK and ORNL. http://www.netlib.org/, último acesso em Janeiro de 2010.

OR-Library (2010). Welcome to OR-Library. http://people.brunel.ac.uk/~mastjjb/jeb/info.html, último acesso em Março de 2010.

Rardin, R. L. and Uzsoy, R. (2001). Experimental Evaluation of Heuristic Optimization Algorithms: A Tutorial. *Journal of Heuristics*, 7(3):261–304.

Triola, M. F. (1999). *Introdução à Estatística*. LTC - Livros Técnicos e Científicos Editora S.A., 7 edition.

TSPLib (2010). TSPLIB. http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/, último acesso em Janeiro de 2010.