

SIMHEURÍSTICA PARA PROGRAMAÇÃO DE MÁQUINA ÚNICA COM TEMPOS DE SETUP DEPENDENTES DA SEQUÊNCIA E ESTOCÁSTICOS

Matheus Lopes Bittencourt^a, Clarissa Maria Rodrigues de Oliveira^a, Raphael Kramer^a,
Isis Didier Lins^{a,b}

Universidade Federal de Pernambuco

^aDepartamento de Engenharia de Produção

^bCentro de Estudos e Ensaios em Risco e Modelagem Ambiental

Av. da Arquitetura, s/n – Cidade Universitária, Recife - PE, Brasil, CEP: 50740-550

{matheus.bittencourt, clarissa.maria, raphael.kramer,
isis.lins}@ufpe.br

RESUMO

Este artigo aborda uma variante do problema de sequenciamento de máquina única, que considera tempos de *setup* estocásticos dependentes da sequência objetivando incorporar incertezas presentes em processos reais. Três simheurísticas são implementadas para resolver um problema de sequenciamento enfrentado por uma indústria de sacos industriais. Os métodos implementados utilizam as meta-heurísticas GRASP, *Simulated annealing* e *Multi-Start* para realizar a busca por diferentes soluções, e Simulação Monte Carlo para avaliá-las em diversos cenários. Três instâncias reais de pequeno porte com tempos de *setup* variando conforme uma distribuição normal foram resolvidas e os resultados obtidos por meio das simheurísticas foram compatíveis aos analíticos, obtidos por meio do uso de propriedades estatísticas de variáveis aleatórias normais, logo, as abordagens foram validadas para o caso da distribuição normal. Espera-se que o uso das simheurísticas em sequenciamentos com número reduzido de tarefas e tempos de *setup* estocásticos com distribuições diferentes da normal forneça resultados satisfatórios.

PALAVRAS CHAVE. Simheurística, Sequenciamento, Máquina única, *Setups* dependentes da sequência, *Makespan*, Otimização estocástica.

ABSTRACT

This article approaches a variant of the single-machine sequencing problem, which considers sequence-dependent stochastic setup times in order to incorporate uncertainties present in real processes. Three simheuristics are implemented to solve a sequencing problem faced by an industrial bag industry. The implemented methods use the GRASP, Simulated annealing and Multi-Start meta-heuristics to perform the search for different solutions, and Monte Carlo Simulation to evaluate them in different scenarios. Three small real instances with setup times varying according to a normal distribution were solved and the results obtained through the simheuristics were compatible with the analytical ones, obtained through the use of statistical properties of normal random variables, therefore, the approaches were validated for the case of the normal distribution. It is expected that the use of simheuristics in sequencing with a reduced number of tasks and stochastic setup times with distributions other than normal will provide satisfactory results.

KEYWORDS. Simheuristics, Scheduling, Single machine, Sequence dependent setups, Makespan, Stochastic optimization .

1. Introdução

Um dos problemas mais estudados na pesquisa operacional é o problema de sequenciamento de máquinas, o qual tem como objetivo determinar o sequenciamento ideal para um grupo de atividades, a fim de otimizar um ou mais objetivos [Gonzalez-Neira et al., 2017]. Nesse sentido, para realizar a programação das tarefas, necessita-se entender o ambiente de manufatura da organização, sendo em geral classificados em *máquina única*, *máquinas em paralelo*, *flow shop*, *job shop* e *open shop* [Pinedo, 2016]. Entre esses tipos de manufatura destaca-se o cenário de máquina única, frequentemente modelado pelo problema de sequenciamento em máquina única (PSMU), em que o layout do processo contém uma máquina com apenas um estágio de produção, no qual, por consequência, cada tarefa é processada uma única vez [Ertem et al., 2019].

O PSMU pode ser classificado como determinístico ou estocástico. Para Chang et al. [2016] a versão determinística do problema foi extensivamente estudada nos últimos anos. Entretanto, esse modelo não caracteriza bem os sistemas reais de produção, visto que em situações reais os processos estão sujeitos a incertezas, que influenciam diretamente na qualidade da solução implementada [Ertem et al., 2019]. Diante disso, os parâmetros estocásticos têm sido cada vez mais levados em consideração para solucionar o problema de sequenciamento de máquina única. Por exemplo, Lu et al. [2014] e Chang et al. [2016] utilizam tempos de processamentos estocásticos em seus trabalhos. Já Ertem et al. [2019] representam os tempos de *setup* como estocásticos.

O PSMU com dados estocásticos pode ser entendido como uma generalização do problema, em que um determinado parâmetro não possui um valor fixo, mas sim um valor aleatório e não-negativo gerado por uma distribuição [Lu et al., 2014]. Dessa maneira, a adição de incertezas na modelagem do problema o torna mais prático, proporcionando a elaboração de soluções mais adequadas ao ambiente real [Abu-Marrul et al., 2022]. Assim, as simheurísticas são uma classe de algoritmos de otimização que lidam de maneira eficiente com essas situações, combinando meta-heurísticas com simulações [Juan et al., 2015]. Sendo caracterizada como uma área de pesquisa ascendente na literatura, devido as suas aplicações e resultados que fornecem mais informações estatísticas sobre uma determinada solução, permitindo uma maior análise do risco para o decisor [Abu-Marrul et al., 2022].

Este artigo tem como objetivo implementar três abordagens simheurísticas, utilizando Simulação Monte Carlo alinhada com meta-heurísticas para resolver um problema real de PSMU com *setups* dependentes da sequência. As abordagens propostas serão aplicadas às instâncias reais apresentadas por Barros e Lins [2021], incluindo no presente estudo características estocásticas para os tempos de *setup* seguindo uma distribuição normal, logo, nos resultados as simheurísticas foram validadas com uso de propriedades estatísticas de variáveis aleatórias Normais, no qual conforme uma amostra suficientemente grande obtidas pela simulação, a média do *makespan* se aproxima de uma distribuição normal, portanto, em casos no qual se considera os *setups* orientados por modelos probabilísticos assimétricos ou a incorporação de outras variáveis estocásticas, as simheurísticas serão úteis para simular cenários com incertezas e encontrar soluções adequadas a realidade.

O restante do artigo está organizado da seguinte maneira: a Seção 2 apresenta a descrição do problema de sequenciamento abordado na pesquisa; a Seção 3 contém conceitos relacionados a simheurísticas; a Seção 4 descreve as abordagens de simheurísticas implementadas nesse trabalho; a Seção 5 demonstra os resultados e experimentos computacionais realizados; por fim, a Seção 6 apresenta as conclusões e perspectivas para trabalhos futuros.

2. Descrição do problema

Pinedo [2016] afirma que um dos objetivos do sequenciamento da produção é o de estabelecer prioridades para orientar a produção. Assim, as atividades são ordenadas seguindo algum

critério específico, entre os quais, listam-se o *makespan*, *flow time* e o atraso total como os mais relevantes. Dessa forma, a versão determinística do PSMU visando à minimização do *makespan* (C_{max}), pode ser definida com um conjunto $T = \{1, 2, \dots, n\}$ de tarefas a serem processadas em uma única máquina, em que o tempo de processamento de cada tarefa $t \in T$ é representado por p_t . Admite-se que todas as atividades estão disponíveis no instante zero e que não é possível a interrupção do processamento de tarefas. Diante disso, uma variante bastante estudada desse problema é o PSMU com tempos de *setup* dependentes da sequência (PSMUSDS), no qual os tempos de configuração são determinados como s_{jk} para j e $k \in T$, em que $s_{jk} \neq s_{kj}$, como pode ser observado Figura 1.

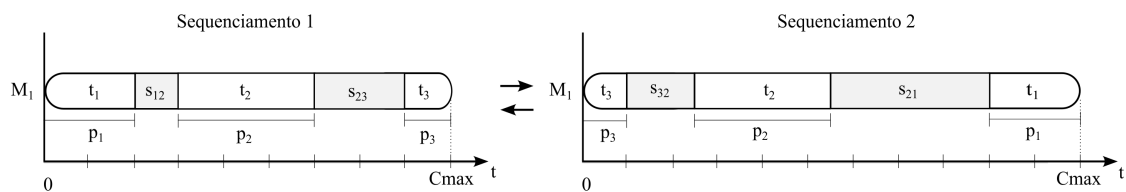


Figura 1: Soluções para PSMU com tempos de *setup* dependentes da sequência.

Assim, como demonstrado no sequenciamento 1 da Figura 1, os tempos de *setup* s_{12} e s_{23} são dependentes tanto das tarefas que já foram produzidas (t_1 e t_2), como das tarefas que ainda serão produzidas (t_2 e t_3), apresentando diferentes tempos para cada combinação de tarefas (concluída e sucessora imediata). Os tempos de processamento das tarefas, por sua vez, podem ser compreendidos como constantes, isto é, independem da ordem em que são processadas. Assim, o que influencia de fato na minimização do *makespan* são as diversas combinações dos tempos de *setup*, que dependem da ordem em que as tarefas são processadas. A incorporação dos *setups* na análise de estudos de sequenciamento é decorrente da sua aplicabilidade em sistemas reais, no qual é possível obter uma enorme economia de tempo e custo quando eles são considerados na modelagem de um ambiente industrial [Allahverdi, 2015]. Logo, ao considerar a variabilidade ou incertezas dos dados e parâmetros do problema, nós temos o *PSMU estocástico*.

O problema abordado neste artigo se refere ao PSMU com *setups* estocásticos dependentes da sequência (PSMUSEDs) que considera as incertezas presentes nos tempos de *setup*, onde esses valores são moldados por uma variável aleatória que segue uma distribuição de probabilidade normal.

3. Simheurísticas

Simheurísticas são estruturas de simulação com otimização que incorporam algoritmos heurísticos e meta-heurísticos para a resolução de problemas com parâmetros estocásticos, obtendo soluções em moderados tempos computacionais [Abu-Marrul et al., 2022]. A combinação de meta-heurísticas com outras metodologias da pesquisa operacional tem se tornado uma alternativa viável para resolver problemas de otimização combinatória com características estocásticas, buscando uma melhor aproximação da realidade, sendo cada vez mais estudada [Gonzalez-Neira et al., 2021].

As simheurísticas têm uma ampla gama de aplicabilidade, desde problemas de sequenciamento como os estudos de Gonzalez-Neira et al. [2021] e Ferone et al. [2016], como também em problemas de roteamento [Keenan et al., 2021], problemas de otimização de portfolio [Kizys et al., 2022], problemas de manutenção [Hilton et al., 2022], entre outros.

Autores como Abu-Marrul et al. [2022]; Ferone et al. [2016]; Gonzalez-Neira e Montoya-Torres [2019] utilizam a Simulação Monte Carlo para gerar diversas combinações de valores para os

parâmetros estocásticos, permitindo uma melhor avaliação das soluções geradas, seja pela obtenção de um valor esperado para a função objetivo, seja para a avaliação do risco e/ou confiabilidade de uma determinada solução [Chang et al., 2016].

Neste trabalho, a Simulação Monte Carlo consiste na geração de valores aleatórios de acordo com a distribuição Normal para a variável estocástica, isto é, para os tempos de *setup* entre pares de tarefas pertencentes a uma dada solução, que é uma sequência fixa de tarefas a serem processadas. Esse processo é repetido um certo número de vezes, de forma a permitir uma simulação de diferentes cenários de desempenho de acordo com as variações dos tempos de *setup*, sendo o *makespan* a função de aptidão analisada. Ao fim, um valor médio para o *makespan* de uma determinada solução é obtido.

As soluções geradas por meio dos procedimentos heurísticos são definidas como promissoras, segundo o pressuposto que soluções com bons desempenhos determinísticos (e.g., considerando os valores de *setup* como os médios) também são boas no ambiente estocástico, em casos de baixa ou média variância [Juan et al., 2015]. A partir disso, caso as mesmas apresentem um *makespan* de qualidade no ambiente determinístico (e.g., soluções geradas a partir da heurística implementada com valor de *makespan* melhor que o melhor já atualizado), são submetidas a Simulações Monte Carlo com uma repetição moderada, de forma a obter uma média que determine previamente se essa solução se comporta bem frente às incertezas. Caso apresente um bom desempenho com base em indicadores estatísticos, esses sequenciamentos promissores são armazenados em um *Pool* para que, ao final da busca no espaço de soluções, sejam submetidas a uma Simulação Monte Carlo com um maior número de repetições, com o objetivo de obter valores esperados de desempenho mais próximos da realidade, permitindo, dessa forma, averiguar a solução que se comporta melhor quando se considera a estocasticidade do problema.

4. Abordagens simheurísticas propostas

Este trabalho apresenta três abordagens com uso de simheurísticas, sendo a primeira e a segunda abordagem uma adaptação da literatura para o problema PSMUSEDs, e a terceira abordagem foi proposta pelos autores do artigo. Em particular, aplicam-se heurísticas construtivas e de refinamento, com base nas meta-heurísticas GRASP [Feo e Resende, 1995], *Simulated Annealing* (SA) [Kirkpatrick et al., 1983] e *Multi-Start*.

4.1. Abordagem com a meta-heurística GRASP

A primeira abordagem para o PSMUSEDs apresentada neste trabalho segue uma estrutura de simheurística associada com a meta-heurística GRASP, pois a mesma tem sido aplicada com sucesso em diferentes trabalhos de sequenciamento com simheurísticas [Gonzalez-Neira et al., 2021; Gonzalez-Neira e Montoya-Torres, 2019; Gonzalez-Neira et al., 2017; Maccarrone et al., 2018; Ferrone et al., 2016]. Além disso, sua implementação é relativamente simples e ela não possui um grande número de parâmetros que requerem longos processos de ajustes e calibração.

A meta-heurística GRASP é caracterizada por um procedimento de construção aleatória e gulosa, seguido por uma fase de busca local. Durante a fase construtiva, uma solução inicial é gerada, adicionando um elemento por vez, de forma iterativa. O elemento a ser adicionado a cada iteração é oriundo de uma lista de candidatos restritos (LCR) que apresenta os maiores benefícios (ou menores prejuízos) à função objetivo. A escolha do elemento ocorre de forma aleatória entre as opções da LCR. Assim, esse artifício auxilia o algoritmo a não ficar preso em ótimos locais [Maccarrone et al., 2018].

No GRASP implementado, utilizou-se a heurística construtiva conhecida como inserção mais barata (IMB), no qual trata de uma estratégia que parte de uma sub-sequência inicial (normalmente composta por três elementos, ou tarefas, escolhidos aleatoriamente) e a cada nova iteração

adiciona uma nova tarefa (ainda não adicionada) entre as tarefas já incluídas na sequência, na posição em que o custo de inserção constitua no mais econômico [Souza, 2022]. Referente ao refinamento, aplicou-se o operador de reinserção (*or-opt*), no qual remove-se um elemento de sua posição e o insere em outra, de maneira a obter um *Best Improvement* [Souza, 2022]. O método é apresentado no fluxograma da Figura 2.

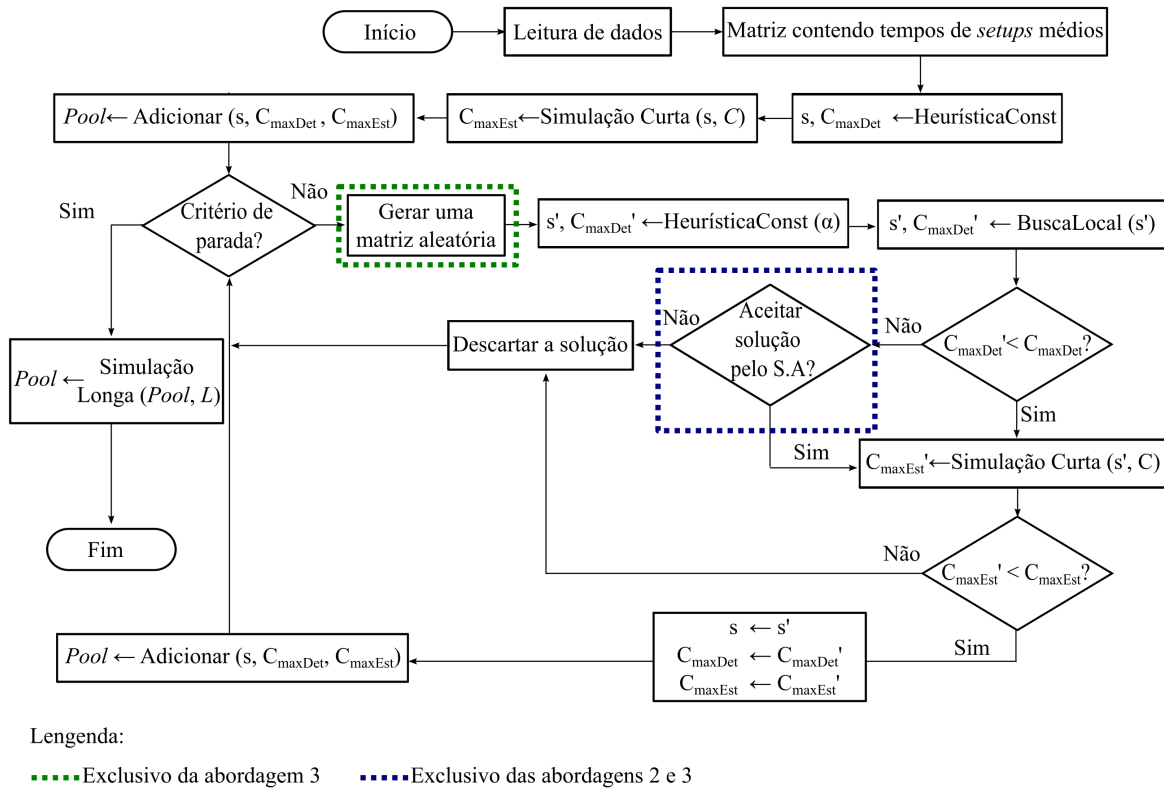


Figura 2: Fluxograma representativo das abordagens 1, 2 e 3.

A solução inicial s é obtida por meio da resolução do problema de maneira determinística (i.e., considerando as médias dos tempos de *setup* para cada par de tarefas). Diante desse sequenciamento, realiza-se uma *simulação curta*, ou seja, com um número de replicações C relativamente pequeno, a fim de obter informações aproximadas sobre o comportamento dessa solução no ambiente estocástico, isto é, o valor do *makespan* esperado. Esse resultado servirá como parâmetro de comparação com as demais soluções encontradas pelo algoritmo. Assim, a solução s é adicionada no *Pool* de soluções promissoras e o melhor valor de *makespan* determinístico e estocástico são atribuídos.

Com essas informações, o algoritmo entra no *loop* da meta-heurística GRASP, que gera novas soluções s' a partir de uma IMB com caráter aleatório devido a LCR, seguido de uma busca local para o refinamento da solução. Diante disso, a solução configura como promissora quando o novo valor gerado é menor que os valores já registrados até o momento, tanto na análise determinística, quanto na estocástica. Em caso positivo, a nova melhor solução (s') é adicionada ao conjunto de soluções promissoras (*Pool*). Após o critério de parada ser atingido (η repetições), o algoritmo parte para a fase final, no qual avaliam-se todas as soluções do *Pool* por meio de uma *simulação longa*, com um número de replicações $L > C$, com a finalidade de obter averiguações

mais incisivas sobre o comportamento estocástico de cada sequenciamento e para selecionar o sequenciamento com menor *makespan* esperado.

4.2. Abordagem com a meta-heurística GRASP e *Simulated Annealing*

A segunda abordagem descrita no presente estudo, verificada na legenda e na Figura 2, emprega um critério de aceitação baseado na meta-heurística *Simulated Annealing* (SA) [Abu-Marrul et al., 2022], o qual tem a finalidade de intensificar a busca por boas soluções, por meio da aceitação de soluções piores do que a melhor já analisada, conforme uma probabilidade, juntamente com a metodologia GRASP para a geração de novas soluções, consoante ao explicado anteriormente.

A meta-heurística SA é desenvolvida com base em uma analogia de resfriamento de átomos aquecidos, no qual o algoritmo busca por melhores soluções implementando uma aleatoriedade na busca e escapando de ótimos locais por meio da aceitação de soluções, mesmo estas não apresentando desempenhos melhores, conforme uma probabilidade definida por $e^{-\frac{\Delta}{T_i}}$ (para função aptidão de minimização), onde Δ é a diferença entre o *makespan* da solução vizinha e o da solução corrente e T_i é um parâmetro associado à temperatura. Ao final de cada iteração a temperatura T_i é reduzida com base em um determinado parâmetro k e, ao atingir um valor de estabilização (e.g., próximo de zero), o procedimento é interrompido [Kirkpatrick et al., 1983]. Portanto, no início do algoritmo (com temperaturas elevadas) há uma maior chance de aceitar soluções piores do que a corrente, permitindo uma diversificação na busca no espaço de soluções. Por outro lado, ao final (com temperaturas baixas) a probabilidade de aceitar soluções piores diminui.

Como observado na Figura 2, verifica-se que a Abordagem 2 difere da anterior no critério de aceitação para execução da *simulação curta* e na condição de parada do algoritmo. Caso a solução retornada pela busca local apresente valor da função objetivo menor que o melhor valor determinístico encontrado até o momento (C_{MaxDet}), a solução é submetida a uma Simulação Monte Carlo curta. Caso contrário, ao invés de descartar a solução de imediato, gera-se um número aleatório $x \in [0, 1]$, com distribuição uniforme, e caso $x \leq e^{-\frac{\Delta}{T_i}}$ a solução é submetida a uma Simulação Monte Carlo curta. Caso contrário, i.e., se $x > e^{-\frac{\Delta}{T_i}}$, descarta-se a solução. A solução corrente s' é adicionada no *Pool* somente se esse valor apresentar comportamento melhor que o melhor já observado no ambiente estocástico; caso contrário, essa solução é descartada. Além disso, o critério de parada é satisfeito quando a temperatura T_i atingir valor inferior ao da temperatura final T_f . Para tanto, ao término de cada iteração, atualiza-se T_i conforme a equação $T_i = T_i \times k$.

4.3. Abordagem *multi-start* e SA com matriz de tempos de *setup* aleatórios

A terceira abordagem, ilustrada na Figura 2, se diferencia das anteriores pelo método utilizado para gerar soluções distintas. As duas abordagens iniciais aplicam uma certa aleatoriedade definida pelo parâmetro $\alpha \neq 0$ do GRASP à heurística construtiva IMB, considerando apenas os dados médios dos *setups*; as variações em desvio são incorporadas somente durante a Simulação Monte Carlo. A terceira abordagem, por sua vez, gera uma matriz de *setups* aleatória a cada iteração (seguindo os parâmetros de média e desvio padrão), conforme demonstrado na legenda da Figura 2. Assim, é possível notar que nessa abordagem as soluções são exploradas já incorporando os desvios previstos nos valores de *setups*.

Diante da Figura 2, percebe-se que, inicialmente, obtém-se uma solução inicial s pelo IMB guloso utilizando os valores médios de *setup*, melhora a solução com uma busca local e, então, calcula o valor de *makespan* determinístico. Então, esse valor de *makespan* (C_{maxDet}) é gravado como o melhor determinístico, o qual é utilizado para definir os parâmetros do SA, correspondentes a temperatura inicial (T_i), temperatura final (T_f) e resfriamento (k). Logo, o melhor

valor estocástico (C_{maxEst}) é determinado após uma Simulação Monte Carlo curta desse sequenciamento em questão, seguindo para o *loop* que realiza a busca por soluções melhores. Uma matriz de *setups* aleatória é gerada, conforme distribuição normal, considerando as médias e os desvios padrão para os tempos de *setups* entre cada par de tarefas, obtendo um sequenciamento por meio da heurística IMB gulosa ($\alpha = 0$), seguida por uma busca local com o operador *reinserção* (*or-opt*). O valor determinístico dessa solução, isto é, o *makespan* considerando as médias dos *setups*, é utilizado para definir as soluções promissoras. Posteriormente, o critério de aceitação do SA funciona conforme o explanado na Abordagem 2.

5. Experimentos Computacionais

O modelo descrito na Seção 2 e os métodos descritos na Seção 3 foram implementados em linguagem de programação Julia, realizando os experimentos computacionais em um computador com processador Intel(R) Core (TM) i5-7200 com CPU de @2.50GHz e 8 GB de memória RAM, com sistema operacional Windows 10. Além disso, foi utilizado o pacote IRace de López-Ibáñez et al. [2016] para calibração automática dos parâmetros dos algoritmos, tomando como base dezoito instâncias de treinamento, as quais foram resolvidas 3000 vezes (*runs of budget*).

As instâncias de treinamento foram geradas aleatoriamente seguindo diferentes configurações de características, relacionadas aos tempos médios e aos desvios padrão dos tempos de *setup*, tempos de processamento e quantidades de tarefas, no qual considerou-se particularidades de número de tarefas pequeno (10) e grande (50), tempos médios de *setup* variando de forma pequena, média e grande, de [50, 70], [100, 200] e [50, 200], assim como para os desvios de [1, 5], [6, 15] e [1, 15], além de tempos de processamento variando no intervalo de [50, 100], resultando em uma combinação de dezoito instâncias.

5.1. Calibração dos parâmetros

Visando obter um melhor desempenho dos algoritmos, os parâmetros C (número de replicações para simulação curta), L (número de replicações para simulação longa), α (controle de aleatoriedade da heurística construtiva) e η (número de replicações do GRASP e do SA) foram calibrados utilizando um intervalo de valores categóricos para delimitar os testes do IRace, como pode ser verificado na Tabela 1, juntamente com a classificação dos grupos, dos itens a serem calibrados das três abordagens, da metodologia aplicada para definição do valor de cada um dos parâmetros e o conjunto de valores retornados pelo IRace para cada abordagem.

Logo, a definição do tamanho do *range* de variação do número de iterações C e L baseou-se nos trabalhos de Maccarrone et al. [2018] e Ferone et al. [2016], para o controle de aleatoriedade α do GRASP foi adotado todo o intervalo possível para o passo de 0.05, e para o η foi adotado o intervalo de 600 até 1500 iterações, visando a não utilizar poucas replicações, para possibilitar o algoritmo ter um caráter mais exploratório, ao mesmo tempo que foi evitado selecionar um valor exorbitante, tendo em vista o custo computacional. Já para a definição dos parâmetros T_i , T_f e k adotou-se a metodologia de Abu-Marrul et al. [2022], no qual os mesmos são calculados e ajustados pelas funções apresentadas na Tabela 1, que dependem de η e C_{MaxDet} , no qual o último corresponde ao *makespan* da primeira solução gerada na inicialização do algoritmo.

5.2. Resultados computacionais

Para avaliar os algoritmos propostos, foram resolvidas as mesmas instâncias consideradas por Barros e Lins [2021], as quais são oriundas de uma coleta de dados de três meses em uma indústria de produção de sacos industriais de papel, o qual apresenta a disponibilidade de 1440 minutos para realizar as tarefas do dia. Em particular, foram consideradas três instâncias (referentes ao dia com maior número de tarefas processadas em cada mês do trimestre avaliado, sendo 7,

Tabela 1: Parâmetros

Grupo	Parâmetros	Variação	Conjunto de Valores		
			Abord 1	Abord 2	Abord 3
Simheurística	C	(100 : 10 : 300)	130	140	160
	L	(1000 : 50 : 2000)	1050	1350	1750
GRASP	α	(0.0 : 0.05 : 1.0)	0.05	0.05	–
	η	(600 : 50 : 1500)	1500	–	–
SA	η	(600 : 50 : 1500)	–	1150	1500
	T_i	–	–	$T_i = \frac{-(0.6 \times C_{maxDet})}{\ln(0.5)}$	
	T_f	–	–	$T_f = \frac{-(10^{-5} \times C_{maxDet})}{\ln(0.5)}$	
	k	–	–	$k = (\frac{T_f}{T_i})^{\frac{1}{\eta}}$	

9 e 9 para janeiro, fevereiro e março, nessa ordem, do ano de 2020). As médias e as variâncias para os tempos de *setup* são reproduzidas a seguir, além dos tempos de processamento das tarefas (Tabela 2). As matrizes têm linhas e colunas relativas aos pedidos (de 1 a 7 na instância 1; de 1 a 9 nas instâncias 2 e 3). Nota-se que, por questões de espaço, excluam-se a linha e a coluna referentes ao pedido fictício 0, necessário para a modelagem constante em Barros e Lins [2021].

$$\begin{aligned}
 E[S^{i1}] &= \begin{bmatrix} 0 & 29 & 45 & 45 & 45 & 70 & 70 \\ 29 & 0 & 45 & 45 & 45 & 70 & 70 \\ 63 & 63 & 0 & 40 & 40 & 55 & 55 \\ 63 & 63 & 40 & 0 & 40 & 55 & 55 \\ 63 & 63 & 40 & 40 & 0 & 55 & 55 \\ 60 & 60 & 55 & 55 & 55 & 0 & 25 \\ 60 & 60 & 55 & 55 & 55 & 25 & 0 \end{bmatrix} & V[S^{i1}] &= \begin{bmatrix} 0 & 4 & 25 & 25 & 25 & 49 & 49 \\ 4 & 0 & 25 & 25 & 25 & 49 & 49 \\ 9 & 9 & 0 & 16 & 16 & 25 & 25 \\ 9 & 9 & 16 & 0 & 16 & 25 & 25 \\ 9 & 9 & 16 & 16 & 0 & 25 & 25 \\ 25 & 25 & 25 & 25 & 25 & 0 & 4 \\ 25 & 25 & 25 & 25 & 25 & 4 & 0 \end{bmatrix} \\
 E[S^{i2}] &= \begin{bmatrix} 0 & 29 & 45 & 45 & 45 & 29 & 68 & 29 & 70 \\ 29 & 0 & 45 & 45 & 45 & 29 & 68 & 29 & 70 \\ 63 & 63 & 0 & 40 & 40 & 63 & 35 & 63 & 55 \\ 63 & 63 & 40 & 0 & 40 & 63 & 35 & 63 & 55 \\ 63 & 63 & 40 & 40 & 0 & 63 & 35 & 63 & 55 \\ 29 & 29 & 45 & 45 & 45 & 0 & 68 & 29 & 70 \\ 97 & 97 & 64 & 64 & 64 & 97 & 0 & 97 & 29 \\ 29 & 29 & 45 & 45 & 45 & 29 & 68 & 0 & 70 \\ 60 & 60 & 55 & 55 & 55 & 60 & 60 & 60 & 0 \end{bmatrix} & V[S^{i2}] &= \begin{bmatrix} 0 & 4 & 25 & 25 & 25 & 4 & 36 & 4 & 49 \\ 4 & 0 & 25 & 25 & 25 & 4 & 36 & 4 & 49 \\ 9 & 9 & 0 & 16 & 16 & 9 & 9 & 9 & 25 \\ 9 & 9 & 16 & 0 & 16 & 9 & 9 & 9 & 25 \\ 9 & 9 & 16 & 16 & 0 & 9 & 9 & 9 & 25 \\ 4 & 4 & 25 & 25 & 25 & 0 & 36 & 4 & 49 \\ 4 & 4 & 36 & 36 & 36 & 4 & 0 & 4 & 4 \\ 4 & 4 & 25 & 25 & 25 & 4 & 36 & 0 & 49 \\ 25 & 25 & 25 & 25 & 25 & 25 & 36 & 25 & 0 \end{bmatrix} \\
 E[S^{i3}] &= \begin{bmatrix} 0 & 98 & 75 & 98 & 98 & 98 & 85 & 83 \\ 88 & 0 & 36 & 20 & 20 & 20 & 121 & 95 \\ 85 & 35 & 0 & 35 & 35 & 35 & 63 & 70 \\ 88 & 20 & 36 & 0 & 20 & 20 & 121 & 95 \\ 88 & 20 & 36 & 20 & 0 & 20 & 121 & 95 \\ 88 & 20 & 36 & 20 & 20 & 0 & 121 & 95 \\ 88 & 20 & 36 & 20 & 20 & 0 & 121 & 95 \\ 92 & 90 & 58 & 90 & 90 & 90 & 0 & 87 \\ 60 & 70 & 80 & 70 & 70 & 70 & 50 & 0 \end{bmatrix} & V[S^{i3}] &= \begin{bmatrix} 0 & 81 & 49 & 81 & 81 & 81 & 81 & 81 & 64 \\ 9 & 0 & 36 & 16 & 16 & 16 & 16 & 121 & 36 \\ 25 & 35 & 0 & 35 & 35 & 35 & 35 & 9 & 49 \\ 9 & 20 & 36 & 0 & 16 & 16 & 16 & 121 & 36 \\ 9 & 20 & 36 & 16 & 0 & 16 & 16 & 121 & 36 \\ 9 & 20 & 36 & 16 & 16 & 0 & 16 & 121 & 36 \\ 9 & 20 & 36 & 16 & 16 & 16 & 0 & 121 & 36 \\ 4 & 81 & 25 & 81 & 81 & 81 & 81 & 0 & 49 \\ 225 & 4 & 9 & 4 & 4 & 4 & 4 & 25 & 0 \end{bmatrix}
 \end{aligned}$$

Tabela 2: Tempos de processamento dos pedidos para cada instância

Instância	P1	P2	P3	P4	P5	P6	P7	P8	P9
$i1$	100	150	275	184	215	70	125	–	–
$i2$	60	55	135	200	160	125	60	230	40
$i3$	277	27	73	78	195	100	174	13	230

Os resultados obtidos foram comparados com aqueles obtidos por Barros e Lins [2021], que fizeram uso de uma abordagem exata com auxílio de um *solver* comercial para resolução de um modelo de programação linear inteira para o PSMUSDS considerando tempos de *setup determinísticos* como sendo a média dos tempos coletados no trimestre mencionado acima, objetivando

minimizar a medida de desempenho *makespan*.

Os resultados computacionais podem ser visualizados na Tabela 3, no qual as sequências de tarefas, valores médios, desvio e intervalo para percentil 5 e 95 obtidos nas simulações para o *makespan* (em minutos), o intervalo analítico (para 90% dos valores de *makespan* em uma distribuição de probabilidade Normal) e os tempos computacionais (em segundos) são apresentados nas colunas Sequência, Média, Desvio, Interv. Simulação, Intervalo Analítico e t_{comp} , respectivamente, para cada abordagem explanada e a exata demonstrada em Barros e Lins [2021]). Vale ressaltar que as sequências apresentada Tabela 3 para o método exato foram obtidas em [Barros e Lins, 2021], logo, a média e desvio correspondente para o *makespan* foram encontrados por meio de uma simulação com 1000 interações obtendo o intervalo para o percentil 5 e 95, além disso, t_{comp} é referente ao tempo gasto para encontrar a solução com a implementação do modelo exato, utilizando o pacote JuMP e CPLEX em linguagem Julia. Cada instância foi resolvida dez vezes, selecionando a solução de melhor *makespan* médio de cada uma.

Tabela 3: Resultados de cada uma das abordagens e método exato para três instâncias apresentadas em Barros e Lins [2021]

Instâncias	Abordagem 1					
	Sequência	Média	Desvio	Interv. da Simulação	Intervalo Analítico	t_{comp}
$i1$	[2, 1, 5, 4, 3, 7, 6]	1352.83	9.25	[1337.33; 1367.63]	[1337.62; 1368.05]	0.18
$i2$	[6, 2, 1, 8, 5, 4, 3, 7, 9]	1340.64	8.87	[1326.42; 1355.32]	[1326.05; 1355.24]	0.37
$i3$	[1, 9, 8, 3, 7, 6, 5, 4, 2]	1422.48	13.50	[1400.18; 1444.48]	[1400.28; 1444.68]	0.42
Instâncias	Abordagem 2					
	Sequência	Média	Desvio	Interv. da Simulação	Intervalo Analítico	t_{comp}
$i1$	[2, 1, 3, 5, 4, 6, 7]	1352.12	9.34	[1337.13; 1367.60]	[1336.76; 1367.48]	0.59
$i2$	[1, 8, 6, 2, 4, 3, 5, 7, 9]	1340.51	8.94	[1325.41; 1355.13]	[1325.79; 1355.22]	1.08
$i3$	[1, 9, 8, 3, 6, 4, 7, 5, 2]	1422.28	13.66	[1399.92; 1444.93]	[1399.81; 1444.75]	0.97
Instâncias	Abordagem 3					
	Sequência	Média	Desvio	Interv. da Simulação	Intervalo Analítico	t_{comp}
$i1$	[2, 1, 5, 3, 4, 6, 7]	1352.41	9.67	[1336.54; 1368.48]	[1336.51; 1368.31]	0.86
$i2$	[1, 2, 6, 8, 3, 5, 4, 7, 9]	1340.53	9.16	[1325.27; 1355.76]	[1325.47; 1355.59]	1.35
$i3$	[1, 9, 8, 3, 4, 2, 7, 5, 6]	1422.33	13.64	[1399.56; 1444.56]	[1399.89; 1444.77]	1.08
Instâncias	Método Exato					
	Sequência	Média	Desvio	Interv. da Simulação	Intervalo Analítico	t_{comp}
$i1$	[1, 2, 3, 4, 5, 6, 7]	1352.77	9.69	[1336.39; 1368.52]	[1336.83; 1368.71]	4.75
$i2$	[2, 1, 8, 6, 3, 4, 5, 7, 9]	1340.84	9.26	[1324.68; 1355.33]	[1325.6; 1356.07]	39.27
$i3$	[1, 9, 8, 3, 5, 4, 6, 2, 7]	1422.46	13.84	[1400.08; 1444.43]	[1399.70; 1445.23]	25.54

Com base na Tabela 3 verifica-se que as melhores médias e desvios obtidos para os *makespans* após as simulações para cada abordagem implementada neste artigo são próximos aos valores médios e de desvio obtidos para as sequências exatas simuladas, apesar das ligeiras mudanças no sequenciamento para cada abordagem. Essas instâncias apresentam múltiplas sequências que levam a um mesmo valor ótimo de *makespan*, por terem pedidos distintos porém da mesma família de produtos. Pedidos de uma mesma família constituem tarefas com a mesma média e o mesmo desvio padrão para os tempos de *setup*; daí a repetição de algumas linhas e colunas, com exceção do valor na diagonal principal, nas matrizes de valores esperados e variâncias apresentadas anteriormente. Já os tempos de processamento (Tabela 2) são todos distintos por conta do tamanho dos lotes de produção. Para a instância $i2$, os pedidos 1, 2, 6 e 8 são de uma família, os pedidos 3, 4 e 5 de uma outra família, o pedido 7 e o pedido 9, cada um pertencente a uma família diferente. Assim, para $i2$, há nove pedidos distribuídos em quatro famílias distintas de produtos (mais detalhes em Barros

e Lins [2021]). Observa-se que, para todas as abordagens, primeiro são processadas as tarefas 1, 2, 6 e 8 (ou permutações delas), em seguida, as tarefas 3, 4 e 5 (ou permutações delas) e, por último, os pedidos 7 e 9 nesta ordem.

Também, observa-se que resultados de qualidade foram alcançados por intermédio de heurísticas com tempos computacionais razoáveis com médias de 0.32s, 0.88s e 1.09s para os métodos 1, 2 e 3, nessa ordem, em contraposição ao resultado do método exato, obtido com 23.18 segundos em média, no qual apesar de corresponder a um tempo satisfatório, vale ressaltar que para quantidades maiores de tarefas (n), essa diferença tende a se tornar cada vez mais significativa. Ademais, como nas simulações considera-se os *setups* normalmente distribuídos, os tempos para *makespan* são distribuídos normalmente também, portanto, tem-se que os intervalos que contêm a média em 90% das vezes, calculados analiticamente, para cada abordagem, são similares aos respectivos intervalos obtidos nas simulações.

Por exemplo, para uma dada sequência fixa, os tempos de *setup* são independentes; note que a dependência dos tempos de *setup* está vinculada à ordem de execução das tarefas e, uma vez definida essa ordem, a fonte de dependência deixa de existir. Por exemplo, considere a sequência ótima encontrada em Barros e Lins [2021] para a segunda instância (referente ao dia com maior quantidade de pedidos em fevereiro de 2020): 2, 1, 8, 6, 3, 4, 5, 7, 9. Se é observado um determinado tempo de *setup* entre os pedidos 2 e 1, essa observação não interferirá na probabilidade de se observar valores de tempo num dado intervalo para a preparação da máquina entre os pedidos 1 e 8.

O *makespan* é a soma dos tempos de *setup*, considerados estocásticos, com os tempos de processamento, supostos determinísticos; portanto, uma combinação linear das variáveis aleatórias independentes que representam os tempos de *setup* e os tempos de processamento. Logo, a média e a variância do *makespan* são definidas pelas equações (1) e (2), respectivamente:

$$E[C_{\max}^{i2}] = E[S_{21}^{i2}] + E[S_{18}^{i2}] + E[S_{86}^{i2}] + E[S_{63}^{i2}] + E[S_{34}^{i2}] + E[S_{45}^{i2}] + E[S_{57}^{i2}] + E[S_{79}^{i2}] + \sum_{t=1}^9 p_t^{i2} \quad (1)$$

$$V[C_{\max}^{i2}] = V[S_{21}^{i2}] + V[S_{18}^{i2}] + V[S_{86}^{i2}] + V[S_{63}^{i2}] + V[S_{34}^{i2}] + V[S_{45}^{i2}] + V[S_{57}^{i2}] + V[S_{79}^{i2}] \quad (2)$$

As propriedades de variáveis aleatórias independentes linearmente combinadas que estão exemplificadas nas equações (1) e (2) são válidas para quaisquer distribuições que caracterizem os comportamentos estocásticos individuais. Ou seja, podemos ter tempos de *setup* independentes regidos por diferentes modelos probabilísticos (Exponencial, Weibull, Lognormal, Normal, entre outros) e, ainda assim, as equações (1) e (2) se verificam (Montgomery e Runger [2018]).

Especificamente nas aplicações deste artigo, cada um dos tempos de *setup* é normalmente distribuído. Assim, pela propriedade reprodutiva da distribuição Normal (Montgomery e Runger [2018]), a distribuição do *makespan* também é Normal. Então, para o exemplo, $C_{\max}^{i2} \sim \mathcal{N}(1341, 82)$ e 90% dos possíveis valores de *makespan* estão no intervalo (1326, 1; 1355, 9). Enfatiza-se que 90% dos valores observados na Simulação Monte Carlo envolvendo a mesma sequência estão no intervalo (1324.68; 1355.33), muito próximo ao obtido analiticamente.

De posse do modelo probabilístico do *makespan*, a operação sabe que é praticamente nula a probabilidade de esse conjunto de pedidos ultrapassar os 1440 minutos disponíveis para a produção. Pode-se então avaliar a inclusão de mais um pedido ao grupo, com o intuito de aproveitar ao máximo o tempo disponibilizado. Nesse caso, é importante reexecutar os modelos considerando o conjunto ampliado de pedidos. Alternativamente, pode-se programar atividades de manutenção para o tempo remanescente. Os resultados das outras instâncias e abordagens podem ser analisados de forma análoga.

6. Conclusão

Esse trabalho implementou três abordagens simheurísticas distintas para a resolução de um problema real de sequenciamento em máquina única apresentado por Barros e Lins [2021], mas com o diferencial de tratar os tempos de *setup* como estocásticos. Assim, o objetivo geral discutido inicialmente foi alcançado com a implementação do primeiro algoritmo combinando Simulação Monte Carlo com a meta-heurística GRASP, do segundo incluindo um critério de aceitação de soluções piores por intermédio do SA e do terceiro consistindo na utilização da meta-heurística Multi-Start gerando matrizes de *setups* aleatórias e realizando simulações.

As simheurísticas foram aplicadas e validadas considerando instâncias reais com tempos de *setup* normalmente distribuídos. Seus resultados foram compatíveis aos analíticos, obtidos por meio do uso de propriedades estatísticas de variáveis aleatórias Normais. Dessa maneira, espera-se que o uso das simheurísticas em sequenciamentos com número reduzido de tarefas (~ 10) e tempos de *setup* estocásticos com distribuições diferentes da Normal (por exemplo, Exponencial, Weibull, Lognormal) forneça resultados satisfatórios, mais próximos à realidade em que Teorema do Limite Central não funciona tão bem. Como tópico de pesquisa corrente, tem-se a aplicação das simheurísticas a contextos com tempos de *setup* regidos por modelos probabilísticos assimétricos.

Ainda, pretende-se realizar um novo estudo incorporando incertezas aos tempos de processamento das tarefas, buscando maiores aproximações com a realidade, além de verificar o comportamento de tais abordagens em instâncias maiores e adaptar as abordagens utilizadas para outros ambientes de manufatura e outras funções objetivo.

Agradecimentos

O presente trabalho foi realizado com apoio da Fundação de Amparo a Ciência e Tecnologia do Estado de Pernambuco (FACEPE), IBPG-1471-3.08/19; do Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq), 404807/2021-6 e 310892/2022-8; e da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES), Código de Financiamento 001.

Referências

- Abu-Marrul, V., Martinelli, R., Hamacher, S., e Gribkovskaia, I. (2022). Simheuristic algorithm for a stochastic parallel machine scheduling problem with periodic re-planning assessment. *Annals of Operations Research*, 320.
- Allahverdi, A. (2015). The third comprehensive survey on scheduling problems with setup times/costs. *European Journal of Operational Research*, 246(2):345–378.
- Barros, G. F. e Lins, I. D. (2021). Sequenciamento ótimo de lotes de produção em uma fábrica de sacos industriais de papel. In *Anais do LVIII Simpósio Brasileiro de Pesquisa Operacional*, João Pessoa - PB.
- Chang, Z., Song, S., Zhang, Y., Ding, J.-Y., Zhang, R., e Chiong, R. (2016). Distributionally robust single machine scheduling with risk aversion. *European Journal of Operational Research*, 256(1):261–274. ISSN 0377-2217.
- Ertem, M., Ozcelik, F., e Saraç, T. (2019). Single machine scheduling problem with stochastic sequence-dependent setup times. *International Journal of Production Research*, 57(10):3273–3289.
- Feo, T. A. e Resende, M. G. (1995). Greedy randomized adaptive search procedures. *Journal of Global Optimization*, 6:109–133.

- Ferone, D., Festa, P., Gruler, A., e Juan, A. A. (2016). Combining simulation with a GRASP metaheuristic for solving the permutation flow-shop problem with stochastic processing times. In *2016 Winter Simulation Conference (WSC)*, p. 2205–2215.
- Gonzalez-Neira, E. M., Ferone, D., Hatami, S., e Juan, A. A. (2017). A biased-randomized simheuristic for the distributed assembly permutation flowshop problem with stochastic processing times. *Simulation Modelling Practice and Theory*, 79:23–36.
- Gonzalez-Neira, E. M., Montoya-Torres, J. R., e Jimenez, J.-F. (2021). A multicriteria simheuristic approach for solving a stochastic permutation flow shop scheduling problem. *Algorithms*, 14(7).
- Gonzalez-Neira, E. M. e Montoya-Torres, J. R. (2019). A simheuristic for bi-objective stochastic permutation flow shop scheduling problem. *Journal of Project Management*, 4:57–80.
- Hilton, S., Langton, J., e Conroy, P. (2022). Digital risk twin driven simheuristic framework for maintenance optimization. In *2022 Annual Reliability and Maintainability Symposium*, p. 1–7.
- Juan, A. A., Faulin, J., Grasman, S. E., Rabe, M., e Figueira, G. (2015). A review of simheuristics: Extending metaheuristics to deal with stochastic combinatorial optimization problems. *Operations Research Perspectives*, 2:62–72.
- Keenan, P., Panadero, J., Juan, A. A., Martí, R., e McGarraghy, S. (2021). A strategic oscillation simheuristic for the time capacitated arc routing problem with stochastic demands. *Computers & Operations Research*, 133:105377.
- Kirkpatrick, S., Gelatt, C. D., e Vecchi, M. P. (1983). Optimization by simulated annealing. *Science*, 220(4598):671–680.
- Kizys, R., Doering, J., Juan, A. A., Polat, O., Calvet, L., e Panadero, J. (2022). A simheuristic algorithm for the portfolio optimization problem with random returns and noisy covariances. *Computers & Operations Research*, 139:105631.
- Lu, C.-C., Lin, S.-W., e Ying, K.-C. (2014). Minimizing worst-case regret of makespan on a single machine with uncertain processing and setup times. *Applied Soft Computing*, 23:144–151.
- López-Ibáñez, M., Dubois-Lacoste, J., Pérez Cáceres, L., Birattari, M., e Stützle, T. (2016). The irace package: Iterated racing for automatic algorithm configuration. *Operations Research Perspectives*, 3:43–58.
- Maccarrone, L., Giovannelli, T., Ferone, D., Panadero, J., e Juan, A. A. (2018). A simheuristic algorithm for solving an integrated resource allocation and scheduling problem. In *2018 Winter Simulation Conference (WSC)*, p. 3340–3351.
- Montgomery, D. C. e Runger, G. C. (2018). *Applied Statistics and Probability for Engineers*. John Wiley & Sons, 7ed edition.
- Pinedo, M. (2016). *Scheduling: Theory, Algorithms, And Systems*, chapter 3rd, p. 33–64. Springer, Berlin, 3rd edition.
- Souza, M. J. F. (2022). Inteligência computacional para otimização: metaheurísticas. Departamento de Computação, Universidade Federal de Ouro Preto, Ouro Preto, Minas Gerais. Disponível em: <http://www.decom.ufop.br/prof/marcone/Disciplinas/InteligenciaComputacional/InteligenciaComputacional.pdf>.