# Tecnologias para a evolução de produtos e serviços: Revisitando o problema da próxima versão

# Technology for the evolution of products and services: Revisiting the next release problem


| | |
|---|---|
| **Marcelo de Souza**<br>https://orcid.org/0000-0002-0786-2127 | Doutor em Computação. Universidade do Estado de Santa Catarina (Udesc) – Brasil. marcelo.desouza@udesc.br. |
| **Maria Cecilia Holler**<br>https://orcid.org/0009-0001-5060-9207 | Graduanda em Engenharia de Software. Universidade do Estado de Santa Catarina (Udesc) – Brasil. maria.holler@edu.udesc.br. |



**RESUMO**

O problema da próxima versão (NRP) visa determinar quais clientes serão atendidos e quais requisitos serão implementados na próxima entrega de um projeto de software. É um problema tradicional da engenharia de software baseada em busca, que se preocupa com o planejamento e evolução de produtos de software. Este trabalho propõe a exploração do NRP em outros domínios, estudando sua aplicação em cenários de tomada de decisão variados. Para isso, foram analisados quatro cenários reais envolvendo a gestão e o planejamento de produtos ou serviços. Foram produzidas instâncias para esses cenários e propostos métodos de solução exato e heurístico. A solução exata se baseia em um modelo de programação linear inteira, resolvido usando um *solver* matemático. Para a solução heurística, é proposto e implementado um algoritmo guloso iterado. Os experimentos computacionais mostram a aplicabilidade do NRP nos cenários estudados, e a eficácia dos métodos propostos. Instâncias pequenas são resolvidas facilmente pela abordagem exata, que é capaz de produzir as soluções ótimas rapidamente. O algoritmo heurístico, por sua vez, é capaz de encontrar as melhores soluções para instâncias grandes em tempo razoável, apresentando desempenho superior ao *solver* nesses casos.



**Palavras-chave:** Otimização. Programação matemática. Metaheurística. Tomada de decisão.



**ABSTRACT**

The Next Release Problem (NRP) aims to determine which customers will be satisfied and which requirements will be implemented in the next release of a software project. It is a traditional problem in search-based software engineering, concerned with the planning and evolution of software products. This work proposes the exploration of the NRP in other domains, studying its application in varied decision-making scenarios. We analyzed four real-world scenarios involving the management and planning of products or services. We produced instances for these scenarios, and proposed exact and heuristic solution methods. The exact solution is based on an integer linear programming model, solved using a mathematical solver. For the heuristic solution, we proposed and implemented an iterated greedy algorithm. The computational experiments show the applicability of the NRP in the scenarios studied, as well as the effectiveness of the proposed methods. Small instances are easily solved by the exact approach, which is capable of quickly producing the optimal solutions. The heuristic algorithm, in turn, is capable of finding the best solutions for large instances in reasonable time, showing superior performance to the solver in these cases.



**Keywords**: Optimization. Mathematical programming. Metaheuristic. Decision making.

Recebido em dd/mm/aaaa. Aprovado em dd/mm/aaaa. Avaliado pelo sistema *double blind peer review*. Publicado conforme normas da xxx.
DOI

## 1 INTRODUÇÃO

A otimização oferece técnicas para solução de problemas de tomada de decisão. Em geral, essas técnicas buscam pela melhor solução (i.e. a melhor decisão) de um problema conforme um objetivo predeterminado e um conjunto de restrições que devem ser satisfeitas. Uma vez que cenários de tomada de decisão são frequentes, a otimização encontra aplicações em variados domínios, como na engenharia, medicina, indústria, agricultura, esportes, entre outros. Algoritmos de otimização permitem a tomada de decisões em cenários complexos, envolvendo muitas variáveis e restrições, o que os torna importantes no planejamento de produtos ou serviços.

O problema da próxima versão (NRP, de *Next Release Problem*) é um exemplo de cenário de tomada de decisão da engenharia de software modelado como um problema de otimização (Bagnall, Rayward-Smith e Whittley, 2001). Considere um software em uso por um conjunto de clientes. Sua evolução tem por base um conjunto de requisitos a serem desenvolvidos, isto é, novas funcionalidades ou características. Para cada nova versão do software, alguns dos requisitos são escolhidos para serem implementados. Essa escolha leva em conta o orçamento para a próxima versão, e.g. a quantidade de horas de trabalho disponíveis, e a satisfação dos clientes. Cada requisito é desejado (ou solicitado) por um subconjunto dos clientes, e cada cliente possui um valor de importância para a empresa, chamado de peso. Um cliente é considerado atendido quando todos os requisitos por ele desejados são implementados. Deseja-se determinar os requisitos que devem compor a próxima versão do software, maximizando o peso total dos clientes atendidos e respeitando o orçamento estabelecido.

Apesar de proposto para planejamento de entregas na engenharia de software, o NRP tem potencial de aplicação em outros contextos envolvendo a evolução de produtos ou serviços. Este trabalho apresenta um estudo da aplicação do NRP em quatro cenários reais diferentes da engenharia de software. Esses cenários foram levantados junto a entidades públicas e privadas, analisando as tomadas de decisão relacionadas à evolução e melhoria dos produtos ou serviços correspondentes. Com base nas características de cada cenário, foram construídas as instâncias que os modelam como NRPs. São propostas duas abordagens para a solução do problema. A primeira consiste na formulação matemática do problema e sua solução exata usando resolvedores (ou *solvers*) matemáticos. A segunda abordagem consiste na solução aproximada do problema usando um algoritmo guloso iterado (IG, de *iterated greedy*) proposto. Um estudo experimental mostra a eficácia dos métodos propostos como ferramentas de suporte à tomada de decisões para evolução de produtos e serviços. Pode-se verificar que as decisões baseadas nas tecnologias propostas permitem a tomada de decisões mais acertadas, conferindo vantagem estratégica nos cenários estudados.

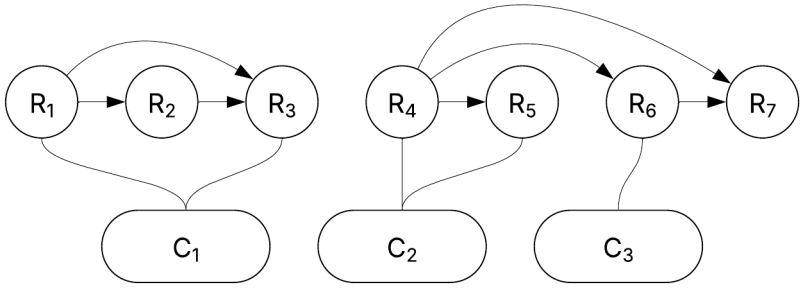
O restante deste trabalho está organizado da seguinte forma. A Seção 2 apresenta o problema da próxima versão e sua formulação matemática. A Seção 3 descreve os cenários de aplicação deste trabalho, seus detalhes e as instâncias construídas a partir das suas características. A Seção 4 descreve o algoritmo guloso iterado, responsável pela solução heurística do problema. A Seção 5 apresenta os experimentos computacionais, seus resultados e discussão. Finalmente, a Seção 6 conclui o trabalho.

## 2 PROBLEMA DA PRÓXIMA VERSÃO

O problema da próxima versão (NRP) é formalizado como um programa linear inteiro. Dados $n$ requisitos, $m$ clientes e um orçamento $b$, $c_i$ é o custo de cada requisito $i \in [n]$ e $w_j$ é o peso de cada cliente $j \in [m]$. $Q$ é o conjunto de pares $(i, j)$ onde o requisito $i$ é solicitado pelo cliente $j$. Requisitos podem ter dependências entre si, onde determinado requisito só pode ser implementado se outro requisito também for implementado (e.g. funcionalidade “reagir a comentário” de uma rede social só pode ser implementada se a funcionalidade “comentário” existir). Nesse caso, $P$ é o conjunto de pares $(i, i')$ onde o requisito $i$ é necessário para o requisito $i'$. A Figura 1 apresenta um exemplo ilustrativo do NRP, com requisitos $\{R_1, R_2, R_3, R_4, R_5, R_6, R_7\}$ e clientes $\{C_1, C_2, C_3\}$. Os requisitos solicitados por

um cliente são conectados a ele por uma aresta. Um requisito dependente apresenta um arco em direção ao requisito do qual ele depende. Note que para atender o cliente $C_3$, neste exemplo, é necessário selecionar os requisitos $R_6$ e $R_7$. O requisito $R_6$ é solicitado por $C_3$, já o requisito $R_7$ é necessário para implementar $R_6$.

**Figura 1**
*Exemplo ilustrativo do problema da próxima versão (NRP) com requisitos $R_i$ e clientes $C_j$.*




Seja $x_i$ variáveis de decisão binárias que assumem valor 1 se o requisito $i$ será incluído na próxima versão e valor 0, caso contrário. Seja $y_j$ variáveis de decisão binárias que assumem valor 1 se o cliente $j$ será atendido na próxima versão e valor 0, caso contrário. O NRP é formalizado como

$$
\begin{aligned}
\text{maximiza} \quad & \sum_{j=1}^{m} w_j y_j, \\
\text{sujeito a} \quad & \sum_{i=1}^{n} c_i x_i \leq b, \\
& x_i \geq x_{i'}, && (i, i') \in P, \\
& x_i \geq y_j, && (i, j) \in Q, \\
& x_i, y_j \in \{0, 1\}, && i \in [n], j \in [m].
\end{aligned}
$$

O NRP foi proposto por Bagnall, Rayward-Smith e Whittley (2001) e, desde então, tem sido amplamente explorado no contexto da engenharia de software (Pitangueira et al., 2013; Harman et al., 2012). Veerapen et al. (2015) apresentam um estudo de formulações inteiras para o NRP (incluindo sua versão bi-objetivo) e a solução exata de instâncias de *benchmark* usadas na literatura. O NRP pode ser reduzido ao problema da mochila (Bagnall, Rayward-Smith e Whittley, 2001; Salkin e De Kluyver, 1975) e, portanto, é um problema NP-difícil (Garey e Johnson, 1979), o que motivou o interesse e aplicação de diferentes métodos de solução heurísticos e aproximados ao longo dos últimos anos (Zhang et al., 2011; Pitangueira et al., 2013).

## 3 APLICAÇÕES

Apesar de inicialmente proposto para o planejamento de entregas de software, o NRP tem potencial de aplicação em outros contextos, apesar de até o momento não existirem trabalhos nessa direção. Esta seção apresenta um estudo em diferentes instituições públicas e privadas, analisando cenários de tomada de decisões para evolução de produtos e serviços. Em particular, o estudo envolveu o planejamento da evolução de produtos ou serviços de uma indústria automobilística, uma indústria têxtil, uma unidade básica de saúde e uma fábrica de bebidas. As seções seguintes descrevem esses cenários e como eles foram modelados como NRPs, apresentando detalhes das instâncias construídas a partir deles.

### 3.1 Indústria automobilística

Neste cenário, a empresa precisa definir quais características e melhorias serão incluídas no novo modelo de um veículo (e.g., opções de motor, itens de segurança, bancos de couro, central multimídia, rodas de liga leve, teto solar). Para isso, ela realiza uma pesquisa de mercado com clientes, os quais devem apontar os itens que julgam importantes para um novo modelo do veículo, dado um orçamento predefinido. Esses clientes são avaliados conforme seu perfil, e recebem uma pontuação. A pontuação é maior quando o perfil indica que o clientes considera ou tem grandes chances de interesse na compra do veículo, bem como aqueles que já possuem veículo da marca e mostram satisfação com o produto.

Diante disso, os requisitos são as características e melhorias que a empresa considera incluir no próximo modelo do veículo. Esses requisitos apresentam dependências entre si, em alguns casos. Por exemplo, a troca de marchas no volante (*paddle shift*) só pode ser incluída se o câmbio automático também for incluído. Os clientes são representados pelos participantes da pesquisa de mercado, a qual já determina o peso de cada um conforme seu perfil. O orçamento é definido pela empresa e consiste no custo adicional disponível para melhorias no novo modelo do veículo.

### 3.2 Indústria têxtil

A empresa têxtil estudada produz diferentes peças de roupas para o público geral. Sua maior receita está na venda de calças jeans a distribuidores e lojistas do sul do Brasil. Duas vezes no ano, a empresa lança uma nova coleção com as peças que compõem seu catálogo para o semestre. Existe uma variedade grande de modelos de calça com diferentes características, como cortes, cores, acessórios, etc. Para escolher os modelos da nova coleção, a empresa considera seu histórico recente de vendas, e inclui as peças com maior procura. Para a escolha dos demais modelos, ela consulta sua rede de distribuidores e clientes lojistas.

Neste cenário, os requisitos são os modelos de calça que podem ser incluídos na nova coleção. Cada modelo tem um custo de produção em horas, e a empresa dispõe de uma capacidade de produção mensal, que define o orçamento disponível e deve ser respeitada. Os clientes são ponderados de acordo com seu volume de compra, i.e. clientes que compram mais possuem maior peso. Os modelos de calça também apresentam dependências entre si. Algumas peças consistem em variações de um outro modelo mais simples que serve de base. Neste caso, o modelo só pode ser incluído na coleção se o modelo de base também for incluído, para evitar problemas na produção.

### 3.3 Unidade de saúde

O terceiro cenário estudado se trata de uma unidade básica de saúde que presta atendimento primário à população. Essa unidade presta diversos serviços de saúde, como atendimento médico, odontológico e psicológico, vacinação, testes rápidos e distribuição de medicamentos. A unidade está planejando uma campanha de conscientização da população sobre a importância dos cuidados com a saúde, o que inclui hábitos de alimentação e exercícios físicos, mas também a busca por atendimento especializado em caso de sintomas incomuns.

A unidade de saúde considera diferentes ações nessa campanha, como a disseminação de conteúdos em redes sociais, visitas e palestras em escolas e distribuição de materiais impressos nas ruas. Cada ação é considerada um requisito, para fins de modelagem do problema. Os clientes são os cidadãos vinculados à unidade de saúde com pelo menos um atendimento de saúde nos últimos três anos. Os pesos dos clientes são inversamente proporcionais ao número de atendimentos de saúde realizados no período, pois a unidade entende que esses pacientes têm maior necessidade de conscientização. Um estudo preliminar estimou quais ações devem atingir cada paciente. Por exemplo, a visita às escolas devem atingir crianças e adolescentes, enquanto a distribuição de materiais no comércio local deve atingir adultos com emprego formal.

Uma vez definidos os requisitos (ações), os clientes (cidadãos/pacientes) e a vinculação de requisitos e clientes (ações que atingem o cidadão), foram estimados os custos, em horas, de cada ação no período de campanha. Também foi definido um orçamento, dado pelo número total de horas disponíveis para a campanha. Finalmente, as ações ainda apresentam dependências entre si. Por exemplo, a distribuição de materiais informativos nas escolas depende da inclusão da ação de visita às escolas.

### 3.4 Produção de bebidas

A *kombucha* é uma bebida gaseificada feita com chá, geralmente preto ou verde, fermentada em duas etapas a partir de uma cultura simbiótica de bactérias e leveduras. O sabor da bebida é obtido pela adição frutas, ervas e aromatizantes na segunda fase de fermentação. O quarto cenário estudado neste trabalho é uma empresa especializada na produção de *kombucha*, com o objetivo de planejar quais novos sabores incluir na produção.

A empresa vende a bebida para mercados e restaurantes da região, e conta com vendedores (representantes) que levam o produto a estabelecimentos do estado. Para o planejamento dos novos sabores, esses clientes foram consultados e selecionaram os sabores de sua preferência. Os pesos dos clientes são definidos em função do volume semestral de compra do produto. Cada sabor possui um custo de produção, principalmente vinculado ao custo dos saborizantes adicionados na segunda fermentação. O orçamento da operação é definido como o custo de produção máximo. Por fim, alguns sabores se diferem pela simples adição de alguma especiaria extra, ou um tempo maior de fermentação. Nesses casos, o sabor que consiste em passos adicionais em relação a outro sabor foi considerado dependente dele.

### 3.5 Instâncias

Com base nas características descritas, foram criadas instâncias do NRP para cada cenário estudado. Em cada cenário, foram definidos três valores distintos de orçamento, representando três situações de tomada de decisão (orçamentos limitado, esperado e alto). A Tabela 1 apresenta o conjunto de instâncias e seus detalhes. A instância *auto* modela o planejamento do próximo modelo de veículo da indústria automobilística. Foram definidas 68 características/melhorias e consideradas as opiniões de 1680 clientes. A instância *cloth* modela o planejamento da nova coleção da indústria têxtil, considerando 180 modelos de calça e 630 clientes. A instância *health* modela o planejamento da campanha de conscientização da unidade de saúde. Essa instância inclui 21 ações de campanha e um público-alvo de 862 pacientes. A instância *drink* modela o planejamento de novos sabores na produção de bebidas. São incluídos 73 sabores nesse estudo e a opinião de 577 clientes.

Para cada instância criada, foi também construída uma instância simplificada (identificada como *small*), que inclui um subconjunto de 100 clientes apenas. Com 4 instâncias pequenas e 4 regulares e 3 valores de orçamento para cada uma, o conjunto de dados considerado possui um total de 24 instâncias. Esse conjunto permite avaliar diferentes cenários de aplicação do NRP, bem como diferentes níveis de complexidade ao incluir instâncias simplificadas.

**Tabela 1**
*Instâncias geradas a partir da pesquisa de campo e suas principais características.*

| Instância | Requisitos | Clientes | Orçamento |
|---|---:|---:|---:|
| auto | 68 | 1680 | {184, 307, 430} |
| cloth | 180 | 630 | {460, 766, 1072} |
| health | 21 | 862 | {56, 93, 130} |
| drink | 73 | 577 | {197, 329, 460} |

## 4. ALGORITMO GULOSO ITERADO

O algoritmo guloso iterado (IG, de *iterated greedy*), proposto por Ruiz e Stützle (2007), é um algoritmo heurístico de otimização. Esse algoritmo combina uma fase de destruição parcial da solução, seguida de sua reconstrução e, opcionalmente, de uma busca local. O IG tem apresentado bom desempenho na solução de diferentes problemas de otimização complexos, como *flow shop* permutacional (Ruiz e Stützle, 2007; Framinan e Leisten, 2008), escalonamento de máquinas paralelas (Fanjul-Peyro e Ruiz, 2010), diversidade máxima (Lozano, Molina e García-Martínez, 2011) e escalonamento de trens (Yuan et al., 2008). Süttzle e Ruiz (2018) apresentam uma revisão do algoritmo IG e de aplicações bem sucedidas na solução de diferentes problemas de otimização.

**Algoritmo 1**
*Pseudocódigo do algoritmo IG (iterated greedy)*

```
1  S ← SoluçãoInicial(α, h)
2  S* ← S
3  para i ← 1 até M faça
4          S' ← Destrói(S, d)
5          S' ← Constrói(S', α, h)
6
7          se S' é melhor que S* então
8                  S* ← S'
9          fim se
10
11         S ← Aceita(S', S*)
12 fim para
13 retorna S*
```

O Algoritmo 1 apresenta o pseudocódigo do IG implementado neste trabalho. O algoritmo inicia construindo uma solução inicial (linha 1), que pode ser gerada aleatoriamente ou usando uma heurística construtiva (descrita abaixo). A solução inicial é atribuída como solução incumbente (linha 2), isto é, a melhor solução encontrada até o momento. O algoritmo executa *M* iterações (laço entre as linhas 3 e 12). A cada iteração, a solução é parcialmente destruída e, em seguida, é reconstruída usando a heurística construtiva (linhas 4 e 5). Caso a solução produzida seja melhor que a incumbente, ela a substitui (linhas 7 a 9). Um critério de aceitação define a solução a ser usada na próxima iteração (linha 11). A solução produzida na iteração atual pode ser adotada para a próxima iteração ou a solução incumbente. Ao final, a solução incumbente é retornada (linha 13).

Uma solução é representada pelo conjunto de clientes atendidos, i.e. cujos requisitos solicitados serão implementados. A **heurística construtiva**, usada para gerar a solução inicial e para reconstruir a solução parcialmente destruída, iterativamente adiciona (ou atende) clientes à solução enquanto há orçamento suficiente. A escolha do próximo cliente a ser adicionado na solução leva em conta uma entre três métrica de avaliação propostas, que visa priorizar os clientes ainda não atendidos e guiar a construção. A métrica W leva em conta o peso do cliente. A métrica C considera o custo dos requisitos ainda não atendidos solicitados pelo cliente. A métrica (W/C) é uma combinação das anteriores, definida pela razão entre o peso do cliente e o custo dos requisitos solicitados ainda não atendidos. Essa terceira métrica pode ser interpretada como uma medida de custo-benefício de incluir o cliente na solução.

Uma vez definida a métrica a ser usada (W, C ou W/C), os clientes que não fazem parte da solução podem ser priorizados. Uma construção puramente gulosa sempre seleciona o melhor cliente cujo custo não ultrapasse o orçamento restante, e repete o processo até não haver orçamento para incluir mais nenhum cliente. Para que a construção produza soluções diversas, a heurística seleciona

aleatoriamente um entre os $\alpha \cdot c$ melhores candidatos, conforme a métrica adotada, sendo $c$ a quantidade de clientes ainda não atendidos cujo custo de atendimento não ultrapassa o orçamento restante e $\alpha = [0, 1]$ um parâmetro do algoritmo. A **destruição** consiste em remover aleatoriamente um subconjunto dos clientes atendidos. A quantidade de clientes a remover é dada por $d \cdot c$, para $c$ clientes pertencentes à solução e um parâmetro $d = [0, 1]$.

O algoritmo IG proposto possui quatro parâmetros principais. O parâmetro $h \in \{W, C, W/C\}$ define a métrica usada na heurística construtiva. O parâmetro $s \in \{$aleatório, heurística$\}$ determina se a solução inicial será gerada aleatoriamente ou usando a construção heurística proposta. O parâmetro $A \in \{S', S^*\}$ define o critério de aceitação. Os parâmetros $\alpha$, $d \in [0, 1]$ são os parâmetros da construção e destruição, respectivamente. Note que esses últimos parâmetros controlam a diversificação da busca. Maiores valores de $\alpha$ fazem com que a construção seja mais aleatória, enquanto menores valores de $\alpha$ tornam a construção mais gulosa. Maiores valores de $d$ descartam uma parte maior da solução atual, o que aproxima o algoritmo de um reinício completo após cada iteração.

## 5. EXPERIMENTOS COMPUTACIONAIS

Os experimentos computacionais foram executados em um computador com processador Apple M2 de 2GHz e 8 núcleos de processamento, 8GB de memória RAM e sistema operacional macOS Sequoia 15.2. O *solver* GLPK (GNU Linear Programming Kit) foi adotado para a solução (exata) do modelo matemático. Cada execução do solver matemático e do algoritmo IG foi realizada usando um núcleo único de processamento. O algoritmo IG foi configurado usando a ferramenta irace (López-Ibáñez et al., 2016) com 10.000 execuções. Os melhores valores encontrados para os parâmetros foram $s$ = heurística, $h$ = W/C, $\alpha$ = 0,4227, $d$ = 0,7918 e $A$ = S*. Essa configuração mostra que, para os cenários propostos, o algoritmo IG apresenta bom desempenho ao usar a métrica W/C na heurística construtiva, com certa aleatoriedade na construção (escolhendo aleatoriamente entre os 42% melhores candidatos) e grande intensidade na destruição (removendo quase 80% dos clientes). Essas características explicam a decisão de usar a solução incumbente a cada nova iteração, uma vez que uma pequena parte da solução é mantida para a fase de reconstrução, que tende a produzir soluções com boa diversidade.

A Tabela 2 apresenta os resultados das abordagens exata (*solver* matemático) e heurística (algoritmo IG) para a solução das instâncias pequenas, i.e. aquelas que consideram somente um subconjunto de 100 clientes. O *solver* foi executado com um tempo limite de 600 segundos, enquanto o algoritmo IG foi executado por $M$ = 2000 iterações e 10 replicações, dada sua natureza não determinística. A Tabela 2 mostra, para cada instância, o valor da solução obtida pelo *solver* e o tempo de execução (colunas “valor” e “t”, respectivamente). Para o algoritmo IG, é apresentado o valor da melhor solução encontrada nas 10 replicações (coluna “melhor”), e o valor médio das soluções encontradas (coluna “média”). São apresentados ainda o tempo médio de execução e o tempo médio até o IG encontrar a melhor solução de cada replicação (colunas “t” e “t sol.”, respectivamente). Valores de soluções ótimas são apresentados sublinhados, e valores das melhores soluções encontradas para cada instância são apresentados em negrito.

A abordagem exata é eficaz na solução das instâncias pequenas. A solução ótima é encontrada pelo *solver* para todas elas. As instâncias *auto*, *health* e *drink* são resolvidas em menos de 1 segundo pelo *solver*. As instâncias *cloth*, no entanto, possuem um número maior de requisitos, o que confere maior complexidade e demanda maior tempo do *solver* para a solução. O algoritmo IG

apresentou desempenho satisfatório, encontrando a solução ótima para a maioria das instâncias em pelo menos alguma das replicações (coluna “melhor”). Apesar do algoritmo IG apresentar tempos de execução maiores para a maioria das instâncias, o tempo de execução para as instâncias *cloth* é consideravelmente menor em comparação com o *solver*. Além disso, o tempo até a melhor solução encontrada é similar àquele observado na solução exata das instâncias. Para instâncias pequenas, a abordagem exata é preferida, pois é capaz de produzir as soluções ótimas no tempo disponível.

**Tabela 2**

*Resultados obtidos pelo solver matemático (solução exata, com tempo limite de 600 segundos) e pelo algoritmo IG (solução heurística, com 2000 iterações e 10 replicações) para as **instâncias pequenas**. Valores de soluções ótimas são apresentadas sublinhadas, e os melhores valores encontrados são apresentados em negrito.*

| Instância | Orçamento | *Solver* | | IG | | | |
|---|---|---|---|---|---|---|---|
| | | valor | t [s] | melhor | média | t [s] | t sol. [s] |
| auto (small) | 184 | **982** | 0,2 | **982** | **982,0** | 3,6 | 0,1 |
| | 307 | **1324** | 0,4 | **1324** | 1323,4 | 4,1 | 0,6 |
| | 430 | **1759** | 0,3 | 1755 | 1752,5 | 4,5 | 1,7 |
| cloth (small) | 460 | **419** | 22,4 | **419** | **419,0** | 6,9 | 0,3 |
| | 766 | **652** | 224,8 | **652** | 650,5 | 7,8 | 3,6 |
| | 1072 | **1134** | 337,7 | **1134** | **1134,0** | 10,8 | 0,9 |
| health (small) | 56 | **1572** | 0,1 | 1558 | 1558,0 | 2,5 | 0,1 |
| | 93 | **1942** | 0,1 | **1942** | **1942,0** | 3,0 | 0,1 |
| | 130 | **2303** | 0,1 | 2258 | 2258,0 | 3,1 | 0,1 |
| drink (small) | 197 | **835** | 0,7 | **835** | **835,0** | 3,2 | 0,1 |
| | 329 | **1184** | 0,9 | **1184** | 1181,1 | 3,9 | 2,0 |
| | 460 | **1671** | 0,5 | **1671** | 1648,6 | 4,6 | 0,7 |

A Tabela 3 apresenta os resultados do *solver* matemático e do algoritmo IG para as instâncias completas, i.e. considerando todos os clientes incluídos no estudo. Ao adotar o conjunto completo de clientes, a complexidade de solução aumenta de forma expressiva. Neste caso, o *solver* é capaz de encontrar a solução ótima para somente duas instâncias no tempo limite de execução. Para as demais, no entanto, são produzidas soluções comparáveis com as produzidas pelo método heurístico. O algoritmo IG foi capaz de produzir as melhores soluções para todas as instâncias, exceto para o cenário *health*. Para os demais cenários, o IG teve desempenho superior à abordagem exata, produzindo soluções melhores (inclusive nos valores médios sobre as 10 replicações). O tempo de execução do algoritmo IG foi menor que o tempo limite do *solver* para todas as instâncias, exceto para o cenário *auto*, onde os tempos superaram os 10 minutos (superando os 20 minutos de execução para duas dessas instâncias, inclusive).

Pode-se observar que as abordagens propostas são eficazes na solução dos diferentes cenários de aplicação do NRP. Além disso, os métodos exato e heurístico apresentam desempenho complementar nos cenários estudados. Para as instâncias pequenas, o *solver* permite a solução exata, produzindo as soluções ótimas e demandando baixo tempo de processamento. Para instâncias maiores e, por consequência, mais complexas, o algoritmo IG é capaz de produzir soluções de qualidade em tempo praticável, superando o desempenho do *solver*. Ambas as abordagens, portanto, se mostraram úteis no suporte à tomada de decisões em diferentes cenários reais.

**Tabela 3**

*Resultados obtidos pelo solver matemático (solução exata, com tempo limite de 600 segundos) e pelo algoritmo IG (solução heurística, com 2000 iterações e 10 replicações) para as **instâncias grandes**. Valores de soluções ótimas são apresentadas sublinhadas, e os melhores valores encontrados são apresentados em negrito.*

| Instância | Orçamento | *Solver* | | IG | | | |
|---|---|---|---|---|---|---|---|
| | | valor | t [s] | melhor | média | t [s] | t sol. [s] |
| auto | 184 | 10425 | 600,0 | **11532** | 11376,1 | 625,9 | 222,5 |
| | 307 | 15756 | 600,0 | **16600** | 16157,2 | 1284,5 | 390,5 |
| | 430 | 24683 | 600,0 | **24810** | 23921,1 | 1205,2 | 623,1 |
| cloth | 460 | 1645 | 600,0 | **1977** | 1974,4 | 170,5 | 80,6 |
| | 766 | 2332 | 600,0 | **2987** | 2945,2 | 201,5 | 97,8 |
| | 1072 | 4089 | 600,0 | **5104** | 5070,6 | 293,7 | 160,5 |
| health | 56 | **13558** | 0,1 | 12923 | 12841,3 | 132,2 | 79,3 |
| | 93 | **15526** | 600,0 | 15217 | 15185,2 | 177,8 | 20,3 |
| | 130 | **18303** | 3,2 | 18128 | 18044,2 | 191,1 | 52,6 |
| drink | 197 | 3568 | 600,0 | **3971** | 3968,5 | 90,7 | 22,9 |
| | 329 | 5374 | 600,0 | **5809** | 5785,0 | 105,2 | 28,7 |
| | 460 | 8347 | 600,0 | **8547** | 8370,3 | 129,6 | 23,1 |

## 6. CONSIDERAÇÕES FINAIS

Este trabalho apresenta a aplicação do problema da próxima versão (NRP) em diversos cenários envolvendo a tomada de decisões na evolução de produtos e serviços. Em particular, o NRP foi aplicado em cenários envolvendo as indústrias automobilística e têxtil, serviços de saúde e produção de bebidas. As características e a tomada de decisões desses cenários foram analisadas para a modelagem e construção de instâncias do NRP. Para resolvê-las, foram implementados um modelo de programação linear inteira e um algoritmo guloso iterado (IG), na qualidade de abordagens de solução exata e heurística, respectivamente.

Os resultados obtidos pela avaliação experimental mostrou que diferentes cenários envolvendo o planejamento de produtos ou serviços podem ser modelados via NRP. Por consequência, modelos matemáticos e algoritmos de otimização para o NRP podem ser úteis para suporte à tomada de decisão nesses cenários. De fato, os experimentos mostram a eficácia dos métodos propostos para a solução das instâncias construídas. A exploração do modelo de programação inteira e sua solução usando um *solver* matemático permite resolver de forma exata as instâncias pequenas (i.e. produzindo as soluções ótimas). Para as instâncias grandes, o algoritmo IG é capaz de produzir boas soluções no tempo disponível, com desempenho superior ao método exato. A combinação das duas abordagens se mostrou ideal para a solução dos problemas propostos.

Este trabalho fornece evidências robustas quanto ao potencial e à importância da tecnologia na gestão de negócios. A otimização, em particular, oferece ferramentas úteis para os mais variados domínios, uma vez que busca pelas melhores alternativas a problemas de tomada de decisões. Ao demonstrar a aplicabilidade do NRP em domínios tão distintos, ressalta-se o poder de abstração de modelos de otimização para capturar características de problemas reais complexos, evidenciando o valor estratégico da otimização como ferramenta de apoio à gestão.

Além disso, a combinação de abordagens exatas e heurísticas, demonstrada neste estudo, oferece um caminho promissor para lidar com a complexidade dos desafios empresariais. Enquanto modelos exatos fornecem soluções ótimas para problemas de menor escala, algoritmos heurísticos se mostram úteis para lidar com a intratabilidade computacional de instâncias maiores e mais

realistas. Essa combinação permite que gestores e tomadores de decisão disponham de ferramentas adaptáveis às diferentes dimensões de seus problemas. Em última análise, a aplicação bem-sucedida do NRP nos cenários explorados não apenas valida a sua relevância para o planejamento e evolução de produtos de software, mas também abre novas avenidas para a pesquisa e a prática em outros domínios.

## REFERÊNCIAS


Bagnall, A. J., Rayward-Smith, V. J., & Whittley, I. M. (2001). The next release problem. *Information and software technology*, 43(14), 883–890.

Fanjul-Peyro, L., & Ruiz, R. (2010). Iterated greedy local search methods for unrelated parallel machine scheduling. *European Journal of Operational Research*, 207(1), 55–69.

Framinan, J. M., & Leisten, R. (2008). A multi-objective iterated greedy search for flowshop scheduling with makespan and flowtime criteria. *Or Spectrum*, 30, 787–804.

Garey, M. R., & Johnson, D. S. (1979). Computers and intractability: A guide to the theory of NP-Completeness. *W. H. Freeman*, New York.

Harman, M., McMinn, P., de Souza, J. T., & Yoo, S. (2012). Search Based Software Engineering: Techniques, Taxonomy, Tutorial. *LASER Summer School*, Lecture Notes in Computer Science, vol. 7007, pp. 1–59, Springer, Heidelberg.

López-Ibáñez, M., Dubois-Lacoste, J., Cáceres, L. P., Birattari, M., & Stützle, T. (2016). The irace package: Iterated racing for automatic algorithm configuration. *Operations Research Perspectives*, 3, 43–58.

Lozano, M., Molina, D., & García-Martínez, C. (2011). Iterated greedy for the maximum diversity problem. *European Journal of Operational Research*, 214(1), 31–38.

Pitangueira, A. M., Maciel, R. S. P., Barros, M. O., & Andrade, A. S. (2013). A systematic review of software requirements selection and prioritization using SBSE approaches. *Search Based Software Engineering*, Lecture Notes in Computer Science, vol. 8084, Springer, Berlin Heidelberg, pp. 188–208.

Ruiz, R., & Stützle, T. (2007). A simple and effective iterated greedy algorithm for the permutation flowshop scheduling problem. *European Journal of Operational Research*, 177(3), 2033–2049.

Salkin, H. M., & De Kluyver, C. A. (1975). The knapsack problem: A survey. *Naval Research Logistics Quarterly*, 22(1), 127–144.

Veerapen, N., Ochoa, G., Harman, M., & Burke, E. K. (2015). An integer linear programming approach to the single and bi-objective next release problem. *Information and Software Technology*, 65, 1–13.

Yuan, Z., Fügenschuh, A., Homfeld, H., Balaprakash, P., Stützle, T., & Schoch, M. (2008). Iterated greedy algorithms for a real-world cyclic train scheduling problem. *International Workshop on Hybrid Metaheuristics*, Springer, Berlin, Heidelberg, pp. 102–116.

Zhang, Y., et al. (2011). Comparing the performance of metaheuristics for the analysis of multistakeholder tradeoffs in requirements optimisation. *Information and Software Technology*, 53 (7), 761–773.