UNIVERSIDADE ESTADUAL DO CEARÁ

CENTRO DE CIÊNCIAS E TECNOLOGIA

PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

MESTRADO ACADÊMICO EM CIÊNCIA DA COMPUTAÇÃO

ALTINO DANTAS BASÍLIO NETO

PLANEJAMENTO DE RELEASE BASEADO EM OTIMIZAÇÃO INTERATIVA ATRAVÉS DA FORMALIZAÇÃO DAS PREFERÊNCIAS DO TOMADOR DE DECISÃO

FORTALEZA – CEARÁ

2016

ALTINO DANTAS BASÍLIO NETO

PLANEJAMENTO DE RELEASE BASEADO EM OTIMIZAÇÃO INTERATIVA ATRAVÉS DA FORMALIZAÇÃO DAS PREFERÊNCIAS DO TOMADOR DE DECISÃO

Dissertação apresentada ao Curso de Mestrado Acadêmico em Ciência da Computação do Programa de Pós-Graduação em Ciência da Computação do Centro de Ciências e Tecnologia da Universidade Estadual do Ceará, como requisito parcial à obtenção do título de mestre em Ciência da Computação. Área de Concentração: Ciência da Computação

Orientador: Prof. PhD. Jerffeson Teixeira de Souza

FORTALEZA – CEARÁ

2016

Dados Internacionais de Catalogação na Publicação

Universidade Estadual do Ceará

Sistema de Bibliotecas

Basílio Neto, Altino Dantas Basílio Neto.
Planejamento de Release Baseado em Otimização Interativa Através da Formalização das Preferências do Tomador de Decisão [recurso eletrônico] / Altino Dantas Basílio Neto Basílio Neto. – 216.
1 CD-ROM: il.; 4 ¾ pol.

CD-ROM contendo o arquivo no formato PDF do trabalho acadêmico com 104 folhas, acondicionado em caixa de DVD Slim (19 x 14 cm x 7 mm).

Dissertação (mestrado acadêmico) – Universidade Estadual do Ceará, Centro de Ciências e Tecnologia, Mestrado Acadêmico em Ciência da Computação, Fortaleza, 2016.
Área de concentração: CIÊNCIA DA COMPUTAÇÃO.
Orientação: Prof. Dr. Jerffeson Teixeira de Souza.

1. Planejamento de Release. 2. Algoritmo Genético. 3. Modelagem de Preferências. 4. Engenharia de Software Baseada em Busca. I. Título.

ATA DA OCTOGÉSIMA SEXTA DEFESA PÚBLICA DE DISSERTAÇÃO DE MESTRADO

Ao vigésimo oitavo dia do mês de março de dois mil e dezesseis, no miniauditório do prédio de Pesquisa e Pós-Graduação em Computação, do Mestrado Acadêmico em Ciência da Computação – MACC, realizou-se a sessão pública de defesa da dissertação de **ALTINO DANTAS BASÍLIO NETO**, aluno regularmente matriculado no Mestrado Acadêmico em Ciência da Computação–MACC, intitulada: ***" PLANEJAMENTO DE RELEASE BASEADO EM OTIMIZAÇÃO INTERATIVA ATRAVÉS DA FORMALIZAÇÃO DAS PREFERÊNCIAS DO TOMADOR DE DECISÃO".*** A Banca Examinadora reuniu-se no horário de 10:30h às 12:30 horas, sendo constituída pelos Professores Doutores **Jerffeson Teixeira de Souza (Orientador/UECE); Paulo Henrique Mendes Maia (UECE); Celso Gonçalves Camilo Júnior (UFG).** Inicialmente o mestrando expôs seu trabalho e a seguir foi submetido à arguição pelos membros da Banca, dispondo cada membro de tempo para tal. Finalmente a Banca reuniu-se em separado e concluiu por considerar o mestrando APROVADO, por sua dissertação e sua defesa pública. Eu, **Professor Dr. PhD. Jerffeson Teixeira de Souza**, Orientador e Presidente da Banca, lavrei a presente Ata que será assinada por mim e demais membros da Banca. Fortaleza, 28 de março de 2016.

Prof. Jerffeson Teixeira de Souza
Orientador – UECE

Prof. Paulo Henrique Mendes Maia
(UECE)

Prof. Celso Gonçalves Camilo Júnior
(UFG)

À toda minha família, por ser meu porto seguro e exemplo de dignidade. Em especial a meu avô materno que é minha maior fonte de inspiração humana.

AGRADECIMENTOS

Primeiramente a Deus pela permissão, graça e misericórdia a mim depreendidas ao longo de minha vida. A Ele toda honra e glória.

Aos meus pais, pelo amor, apoio incondicional e sobretudo por serem meus maiores exemplos de dignidade e coragem.

Ao meu avô Altino Dantas pelo nome e a herança de honestidade, ombridade e integridade, valores sem os quais esta conquista não faria sentido.

Obrigado aos meus irmãos e sobrinhos, pessoas maravilhosas que foram por muitas vezes minha válvula de escape, e, nos momentos de minha ausência em função dos estudos, estiveram cientes de que o futuro é feito a partir da constante dedicação no presente.

À esta universidade, seu corpo docente, direção e administração que oportunizaram a janela através da qual, vislumbro um horizonte promissor. À FUNCAP pelo fomento.

A todos os professores por me proporcionarem o conhecimento não apenas racional, mas a manifestação do caráter e afetividade da educação no processo de formação profissional, especialmente aos egrégios mestres Valdísio Viana, Paulo Henrique Maia e Leonardo Sampaio.

Ao meu orientador professor Jerffeson Teixeira por toda instrução técnica e principalmente pelos demais conhecimentos que certamente servir-me-ão por toda vida.

Aos membros da célula Mais de Deus que me acolheram e aturaram com muito carinho. Especial agradecimento à Jéssica Almeida e Letícia Linhares pelas mais que agradáveis companhias em diversos momentos, ao Felipe Barreto pelas conversas e exemplo de dedicação e ao Alberto Ribeiro pelo cuidado.

Ao Pr. José de Castro Gomes e família que me abrigaram e por mim fizeram muito mais do que se poderia imaginar. A família Carrara que me acolheu de tal forma que jamais poderei recompensá-los.

À Juliana Machado pela motivação inicial, à Lia Reck pelas dicas e a todos os amigos preciosos que direta e indiretamente estão ou estiveram presentes no decorrer dessa etapa.

Ao Allysson Allex pelo distinto companheirismo, ao Italo Yeltsin pelo incansável auxílio técnico, ao Duany Dreyton, Domingos Sávio e Lucas Roque pelas experiências compartilhadas, ao Thiago Nascimento e Matheus Paixão pelos exemplos de competência, conversas e orientações valiosas. Por fim, a todos os membros do GOES e alunos do MACC que foram influências edificadoras nestes últimos dois anos.

“Porque estou certo de que, nem a morte, nem a vida, nem os anjos, nem os principados, nem as potestades, nem o presente, nem o porvir, nem a altura, nem a profundidade, nem alguma outra criatura nos poderá separar do amor de Deus, que está em Cristo Jesus nosso Senhor.”

(Apóstolo Paulo)

RESUMO

A Engenharia de Software (ES) enfrenta desafios relacionados aos métodos e processos que envolvem a construção do seu produto alvo, o software. Por sua vez, a Engenharia de Software Baseada em Busca, do inglês, *Search-based Software Engineering* (SBSE) é um campo de pesquisa voltado à resolução de problemas complexos da ES, através da aplicação de técnicas de Otimização Matemática. Nesse escopo, destaca-se o Planejamento de *Releases* (PR), que é uma tarefa complexa e envolve diversos aspectos relacionados à decisão de quais requisitos devem ser implementados em cada versão de um software construído de forma iterativa e incremental. Muitos algoritmos de otimização têm sido aplicados ao PR, todavia, na maioria deles a expertise do Tomador de Decisão não é efetivamente considerada para a geração de soluções, o que pode demonstrar-se como inadequado no tocante a aceitação dos resultados. Em razão disso, o presente trabalho propõe uma abordagem para resolução do PR, através da aplicação de Otimização Interativa apoiada por uma base de preferências dinamicamente modificada pelo Tomador de Decisão, de modo que as soluções encontradas contemplem tanto os aspectos próprios do PR quanto as convicções do referido profissional. Para isso, foi estabelecido um modelo para definição de tipos de preferências, a partir do qual formalizou-se 8 tipos de preferências relativas à opinião do Tomador de Decisão em relação à distribuição dos requisitos em *releases*. Dessa forma, elaborou-se uma formulação interativa para o referido problema e adaptou-se um Algoritmo Genético para geração das soluções. Considerando instâncias reais e artificiais, a abordagem foi avaliada tanto por simulação das interações humanas quanto por profissionais com experiência em desenvolvimento de software. Demonstrou-se, empiricamente, que a proposta consegue priorizar a satisfação das preferências mais importantes, e, em alguns casos satisfaz todas as preferências, perdendo não mais do que 11% na otimização dos aspectos próprios do PR. Ademais, a abordagem proposta é capaz de aumentar a satisfação do Tomador de Decisão, em média, de 44% para 82% quando comparada a uma abordagem não interativa.

Palavras-chave: Planejamento de *Release*. Algoritmo Genético. Modelagem de Preferências. Engenharia de Software Baseada em Busca.

ABSTRACT

Software Engineering (SE) has a lot of challenges regarding to the processes and methods used to build its target product. In turn, Search-based Software Engineering (SBSE) is a search area inclined to solve SE complex problems through Mathematic Optimization techniques. In this scope, it is highlighted Release Planning Problem (RP) that is a complex task which evolves several aspects related to decision of which requirement most be implemented in each version of a software iteratively and incremental developed. Different optimization algorithms have been applied to tackle this problem, however, most of their don't consider the human expertise to generate solutions, so, this can be inappropriate with respect to acceptance of the results. Thus, this work proposes an approach to solve the Release Planning Problem by usage Interactive Optimization supported by a preference base which is dynamically updated by Decision Maker, so that, found solutions contemplate as RP aspect as convictions of this professional. To enable this idea, a model to definition of preferences type was introduced, from which 8 types of preferences related with the Decision Maker opinion on requirements assign were formal defined. A interactive formulation to the cited problem was elaborated and a Genetic Algorithm was adapted to generate solutions. Considering real and artificial instances, the proposed approach was evaluated by simulations of human interactions and by expert professionals in software development area. The empirical study shown the proposal is able to prioritize the satisfaction of most important preferences and, sometimes, satisfy all preferences, losing no more than 11% on optimization of RP intrinsic aspects. In addition, the proposed approach is capable to increase the DM satisfaction, in average, from 44% to 82% when it is compared with a non-interactive approach.

Keywords: Release Planning. Genetic Algorithm. Preferences Modeling. Search-Based Software Engineering.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE QUADROS

LISTA DE ALGORITMOS

LISTA DE ABREVIATURAS E SIGLAS

ACO	*Ant Colony Optimization*
AG	Algoritmo Genético
AGI	Algoritmo Genético Interativo
CE	Computação Evolucionária
CEI	Computação Evolucionária Interativa
DP	Densidade de Preferências
ES	Engenharia de Software
GP	Ganho em Preferências
NRP	*Next Release Problem*
NS	Nível de Satisfação
PP	Preço das Preferências
PR	Planejamento de *Releases*
PS	Preferências Satisfeitas
SBSE	*Search-based Software Engineering*

LISTA DE SÍMBOLOS

α	Nível de significância em testes estatísticos;
λ	Importância relativa de cada *stakeholder*;
μ	Parâmetro que define o peso da influência da Base de Preferências durante o processo de busca;
Ω	Conjunto que representa o espaço de busca para um problema de otimização;
ξ	Importância relativa a cada *release*;
$\triangle$	Diferença estatística insignificantemente maior;
$\triangledown$	Diferença estatística insignificantemente menor;
$\blacktriangle$	Diferença estatística significantemente maior;
$\blacktriangledown$	Diferença estatística significantemente menor.

SUMÁRIO

1 INTRODUÇÃO

A Engenharia de Software (ES) é uma disciplina de engenharia relacionada com todos os aspectos da produção de software, desde os estágios iniciais de especificação do sistema até a sua manutenção, ou seja, mesmo após este entrar em operação (SOMMERVILLE, 2011). Naturalmente, a ES descreve diversos modelos de desenvolvimento de software, dentre eles, o modelo iterativo e incremental, no qual o produto é desenvolvido e disponibilizado para o cliente em partes funcionais denominadas *releases*.

A decisão de quais requisitos devem ser implementados em cada *release* é uma das tarefas complexas inerentes ao desenvolvimento interativo e incremental de software. O Planejamento de *Releases* (PR) é reconhecidamente um problema cognitiva e computacionalmente complexo (NGO-THE; RUHE, 2008). Tal problema envolve muitos aspectos, tais como as necessidades e/ou desejos dos interessados no produto e as restrições peculiares às características das funcionalidades a serem implementadas. Idealmente, o Planejamento de *Releases* deve maximizar o valor de negócio através da melhor sequência de entregas, satisfazer os *Stakeholders*[1] mais importantes, contemplar os recursos disponíveis e refletir as dependências entre os requisitos (RUHE; SALIU, 2005). O desafio é que, além de dinâmicos, esses objetivos são muitas vezes conflitantes.

Problemas complexos da Engenharia de Software têm sido abordados com técnicas computacionais que visam automatizar suas resoluções, e esse tipo de estratégia foi definido por Harman e Jones (2001) como Engenharia de Software Baseada em Busca, do inglês *Search-based Software Engineering* (SBSE). Assim, a SBSE é recente campo de pesquisa que explora problemas relacionados à diversos aspectos da ES, tais como, Engenharia de Requisitos, Projeto e Gerenciamento ou Testes e Manutenção, reformulando-os como problemas de Otimização Matemática e resolvendo-os através com aplicação de algoritmos de busca.

1.1 MOTIVAÇÃO

Em seu trabalho, Souza *et al.* (2010) demonstraram que a aplicação de técnicas de busca pode superar os resultado obtidos por humanos para resolver tarefas complexas. Além disso, Bagnall, Rayward-Smith e Whittley (2001), Baker *et al.* (2006) e Zhang, Harman e Mansouri (2007) mostraram que é viável a aplicação dos conceitos de SBSE no âmbito da

[1] O termo *Stakeholder* refere-se às pessoas envolvidas no processo de desenvolvimento do software ou interessadas no produto final.

Engenharia de Requisitos. Todavia, a maioria das atuais abordagens de SBSE para planejamento de *releases* de software não considera de forma efetiva o conhecimento do tomador de decisão [2] durante o processo de geração das soluções. Por isso, esse profissional pode demonstrar um certo nível de rejeição aos resultados obtidos, visto que sua expertise não fora devidamente capturada durante o processo de decisão (MIETTINEN, 2012).

Além da presumida rejeição aos resultados obtidos de forma automática, é importante destacar que o Planejamento de *Releases* é caracterizado por envolver diversos fatores subjetivos e inerentemente dinâmicos, os quais concorrem para uma boa qualidade da solução final, mas são difíceis ou impossíveis de serem modelados matematicamente (NGO-THE; RUHE, 2008).

Alguns trabalhos limitam-se em incorporar a subjetividade dos envolvidos no projeto através da atribuição de notas (valores de importância) ou indicação da *release* preferível para cada requisito. Nesses casos, as informações são valores de entrada dos algoritmos e permanecem imutáveis durante todo o processo de busca.

A presença da subjetividade, a incapacidade de modelar alguns fatores importantes ou a ausência de informações suficientes para guiar a busca por soluções, têm sido justificativas para a aplicação de Otimização Interativa em SBSE, o que pode ser percebido nas pesquisas de Tonella, Susi e Palma (2010), Bavota *et al.* (2012) e Araújo e Paixão (2014). A escolha de tal estratégia explica-se pela capacidade dela em capturar informações provenientes do usuário e refleti-las na solução final gerada pelos algoritmos de busca. Todavia, o tipo de opinião requerida do usuário e, consequentemente, como esta será capturada, representam importantes desafio à aplicação desta técnica.

A partir do levantamento bibliográfico realizado, foi percebido que apesar de ser um problema bastante explorado em SBSE e pesquisas apontarem a importância da expertise humana no contexto da engenharia de requisitos, até o momento não verificam-se trabalhos que explorem a modelagem matemática da opinião do tomador de decisão durante o processo de geração de soluções para o PR.

[2] Esta é uma tradução livre do termo *Decision Maker* amplamente utilizado na literatura. Neste trabalho, tomador de decisão é o profissional responsável por definir a distribuições das funcionalidades em *releases*, seja ele um gerente de projetos, engenheiro de requisitos ou qualquer outro profissional com tal responsabilidade.

1.2 OBJETIVOS

1.2.1 Objetivo Geral

Diante do exposto, o objetivo principal desse trabalho é propor e desenvolver uma abordagem para resolução do problema de Planejamento de *Releases* (PR), através da aplicação de Otimização Interativa apoiada pela definição formal das preferências do tomador de decisão com relação à alocação dos requisitos em *releases*, de modo que as soluções geradas possam considerar tanto os aspectos próprios do PR quanto as convicções do referido profissional.

1.2.2 Objetivos Específicos

Com a finalidade de atingir o objetivo principal deste trabalho, alguns objetivos específicos foram estabelecidos e encontram-se descritos a seguir:

a) levantar e modelar matematicamente os tipos de preferências que o tomador de decisão pode ter com relação à distribuição das funcionalidades durante o planejamento das *Releases* de um software;
b) elaborar uma formulação interativa para o problema do Planejamento de *Releases* de Software;
c) implementar um Algoritmo Genético que considere as preferências humanas durante o processo de geração das soluções;
d) implementar uma interface de interação através da qual o usuário possa manipular suas preferências;
e) testar e analisar o comportamento da abordagem quanto a(ao):
   - influência das preferências nas soluções geradas pelo algoritmo de busca;
   - consequência da satisfação das preferências;
   - capacidade de priorização das preferências mais importantes;
   - relação entre a quantidade de preferências expressadas pelo usuário e o percentual de atendimentos destas;
   - satisfação de um tomador de decisão diante de uma solução obtida considerando-se a intervenção do mesmo;
   - relação entre o nível de satisfação das preferências mais importantes e o sentimento de satisfação do tomador de decisão.

1.3 ORGANIZAÇÃO DO TRABALHO

Além da presente Introdução, este trabalho é organizado conforme a seguir.

No Capítulo 2 apresenta-se o referencial teórico que fundamenta este trabalho. Inicialmente são apresentados os princípios do campo da Otimização Matemática com indicação das principais técnicas de busca. Em seguida mostra-se a Otimização Interativa, com suas características e principais desafios. Posteriormente, é descrito o funcionamento dos Algoritmos Genéticos. Logo após, descreve-se como problemas da ES podem ser modelados como problemas de busca. Além disso, são apresentados os aspectos que caracterizam o Planejamento de *Releases* e explicado como este problema pode ser resolvido com técnicas de otimização. Por fim, destaca-se a aplicação de análises estatísticas de experimentos em SBSE.

O Capítulo 3 trás os principais trabalhos relacionados às ideias centrais desta pesquisa. São apresentados trabalhos que abordam o PR através da aplicação de meta-heurísticas. Em seguida, são apresentadas pesquisas relativas ao contexto de Engenharia de Requisitos que fazem da Otimização Interativa e por fim, são discutidos trabalhos que, como este, usam Computação Evolucionária Interativa tendo as preferências do usuário expressadas de forma explícita.

O Capítulo 4 apresenta a abordagem proposta por este trabalho para o Planejamento de *Releases*. Na primeira seção é mostrada uma visão geral do funcionamento da técnica e dos elementos que a compõem. Em seguida, é descrita a modelagem adotada para o PR, destacando-se os detalhes relativos a requisitos, *releases*, clientes e representação da solução. Depois é definido como a opinião do tomador de decisão é modelada, e por fim, exposta a formulação matemática interativa do referido problema.

No Capítulo 5 encontram-se os detalhes à cerca do estudo empírico conduzido para avaliação da abordagem proposta. Inicialmente, apresentam-se as questões de pesquisa que nortearam os experimentos e as configurações e procedimentos metodológicos utilizados. Em seguida, são apresentados os resultados obtidos bem como a análise destes através de tabelas e gráficos. Por fim são descritas algumas ameaças à validade dos experimentos.

Finalmente no Capítulo 6 são apresentadas as considerações finais a cerca do trabalho, bem como as perspectivas de evoluções da atual pesquisa através da previsão de trabalhos futuros.

2 FUNDAMENTAÇÃO TEÓRICA

Neste capítulo será apresentado o referencial teórico que fundamenta este trabalho. Inicialmente são apresentados os conceitos primordiais do campo de Otimização Matemática com destaque para as técnicas de busca mais conhecidas na literatura. Em seguida são relatadas as definições de Otimização Interativa, comentando-se suas características e principais desafios. Posteriormente, é mostrado um detalhamento dos elementos e funcionamento dos Algoritmos Genéticos. Logo após, há uma descrição de como problemas da Engenharia de Software são modelados como problemas de busca. Além disso, são apresentados os aspectos que caracterizam o Planejamento de *Releases* e explicado como este problema pode ser resolvido com técnicas de otimização. Por fim, destaca-se como podem ser conduzidas análises estatísticas no contexto dos experimentos em SBSE.

2.1 OTIMIZAÇÃO MATEMÁTICA

A Otimização Matemática é um campo de conhecimento cujas técnicas visam determinar os extremos (máximos ou mínimos) de funções em determinados domínios. De maneira concreta, o extremo que se deseja descobrir pode representar algum fator de mérito ou qualidade de um sistema que se deseja construir ou analisar (CUNHA; TAKAHASHI; ANTUNES, 2012).

Para Andrade (1998), determinar os valores extremos de uma função, buscando encontrar o maior ou o menor valor que esta pode assumir em um dado intervalo, é um Problema de Otimização. Assim sendo, no mundo real existem diversas situações modeláveis como problemas de otimização. Por exemplo, a definição de como deve ser cortada uma peça de tecido para confecção de vários itens de vestuário, de modo a minimizar a perda do tecido. Outro caso poderia ser a escolha de qual rota um caminhão de recolhimento de lixo precisa fazer, se a meta for “visitar” a maior quantidade de pontos da cidade no menor tempo possível.

De modo geral, os problemas do mundo real são modeladas como problemas de otimização definindo-se uma Função Objetivo que, contemplando as variáveis do problema, possa matematicamente representar o objetivo a ser atingido. Admitindo que exista um espaço contendo todas as possíveis soluções para o problema, denominado espaço de busca, uma dada configuração das variáveis faz com que a função objetivo assuma um determinado ponto nesse espaço. Dessa forma, resolve-se o problema encontrando a combinação de valores das variáveis

que produza o valor ótimo, isto é, um dos extremos da função.

A despeito dos objetivos poderem ser tanto de maximização (encontrar o ponto máximo da função objetivo) quanto de minimização (descobrir o ponto mínimo), os modelos são tipicamente expressos na forma de minimização seguindo a fórmula geral abaixo:

$$\begin{aligned} \text{mininizar} \quad & f(x), \\ \text{sujeito a:} \quad & g_i(x) \leq 0 \; para \; i = \{1,2,3,\cdots,m\}, \\ & h_j(x) = 0 \; para \; j = \{1,2,3,...,p\}, \\ & x \in \Omega, \end{aligned}$$

onde $f(x)$ é a função para a qual se deseja encontrar um valor ótimo, $g(x)$ e $h(x)$ representam as restrições do problema e Ω é o conjunto de todas as soluções existentes. A variável x representa uma solução do conjunto Ω, a qual deve respeitar todas as restrições do problema para ser considerada válida ou factível.

As características dos elementos que compõem a modelagem de um problema, o que de certa forma reflete a natureza deste, podem caracterizar tipos especiais de otimização. Um exemplo é a chamada Otimização Combinatória, a qual trata problemas de otimização que possuem apenas variáveis discretas (CUNHA; TAKAHASHI; ANTUNES, 2012). Diante da grande quantidade e variabilidade de problemas práticos de otimização, diversas técnicas de busca têm sido estudadas e aprimoradas com intuito de se explorar eficientemente tais problemas. De forma geral, as técnicas de busca estão intimamente ligadas às características do problema e tipicamente se diferenciam quanto as estratégias de exploração do espaço de busca.

Os chamados métodos exatos são técnicas de otimização bastante difundidas. Como o próprio nome sugere, tratam-se de algoritmos que possuem a capacidade de encontrar de forma precisa, soluções ótimas para um determinado problema, a partir de uma sequência finita de passos. Todavia, a aplicação destes métodos é condicionada a existência de determinadas propriedades matemáticas na função objetivo, nas variáveis e nas restrições definidas na modelagem do problema. Além disso, as técnicas exatas demandam grande esforço computacional o que eventualmente, dependendo da instância do problema, torna inviável a utilização de tais métodos.

Alternativamente aos métodos exatos, existem as meta-heurísticas, que são técnicas heurísticas de propósito geral, amplamente empregadas para resolução de problemas de otimização combinatória (CUNHA; TAKAHASHI; ANTUNES, 2012). Tais métodos não garantem a descoberta da solução ótima, mas na prática encontram uma solução suficientemente boa. Dessa

forma, as meta-heurísticas são robustas ao ponto de serem aplicadas à qualquer problema de otimização, ao custo de pontuais adaptações em suas estruturas típicas.

Enquanto os métodos exatos são determinísticos, isto é, diferentes execuções para uma mesma instância sempre retornam o mesmo resultado, as meta-heurísticas possuem caráter estocástico e, por isso, execuções distintas tendem a produzir resultados diferentes.

Dentre as meta-heurísticas mais utilizadas podem ser destacados os Algoritmos Genéticos, Otimização por Colônia de Formiga, do inglês *Ant Colony Optimization* (ACO), Busca Tabu ou *Tabu Search* (TS) e Têmpera Simulada ou *Simulated Annealing* (SA) (GENDREAU; POTVIN, 2010).

2.2 OTIMIZAÇÃO INTERATIVA

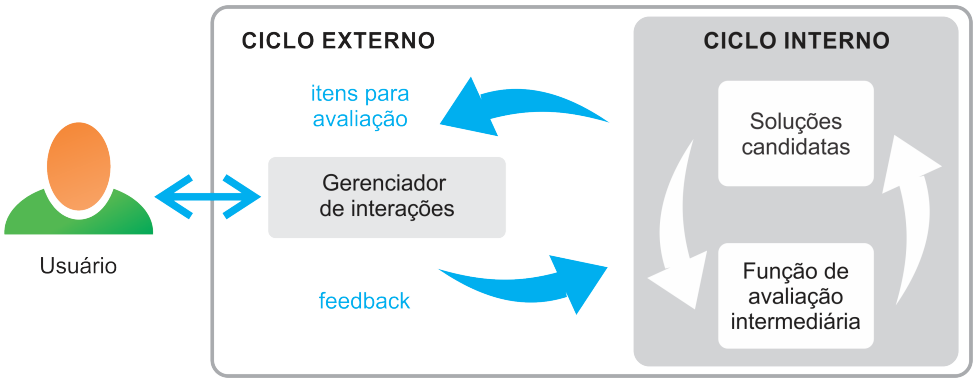
Existem cenários de otimização nos quais não é simples definir uma função objetivo que seja suficientemente capaz de capturar as informações necessárias para explorar todas as áreas do espaço de busca. Em muitos desses casos, o conhecimento e capacidade de efetuar julgamentos, naturais dos seres humanos, podem ser utilizados para auxiliar a técnica responsável pelo processo de busca, logo, surge assim, a ideia de Otimização Interativa.

Para Fisher (1986) a otimização interativa significa aproveitar a máquina para desempenhar as funções nas quais ela é superior, e o ser humano naquilo que ele leva vantagem. De forma mais específica, Miettinen (2012) define que nos métodos interativos o tomador de decisão trabalha iterativamente em conjunto com o algoritmo até que se chegue a uma solução considerada boa o suficiente pelo usuário. Para que isso seja possível, a técnica computacional deve produzir informações, enquanto o ser humano tem a responsabilidade de avaliá-las e fornecer algum *feedback*. Finalmente, os dados advindos do usuário precisam ser utilizados pelo algoritmo a fim de que novas soluções sejam geradas de modo a incorporar as preferências humana.

A Figura 1 mostra um modelo de otimização interativa proposto por Marculescu, Feldt e Torkar (2012) para geração de casos de teste. Nele, há um "ciclo interno" onde acontece o processo de otimização e um "ciclo externo" concernente à interação usuário-máquina.

Harman *et al.* (2012) apontam que uma das ocasiões oportunas para se utilizar otimização interativa é quando existe naturalmente algum caráter subjetivo que distingua as soluções de um problema. Nesses casos, a avaliação das soluções pode ser total ou parcialmente realizada por critérios de julgamento humano. Já em outras situações, como por exemplo no

Figura 1 – Exemplo de abordagem com Otimização Interativa



Fonte: Adaptado de Marculescu, Feldt e Torkar (2012).

estudo de Thiele *et al.* (2009), o motivo para incluir o ser humano no processo de busca é concentrar os esforços de exploração em áreas que sejam de maior relevância para o usuário.

A necessidade de inclusão da expertise humana durante o processo de otimização promovido pelas técnicas de busca, aliada à ampla disseminação da computação inspirada nos princípios naturais de evolução das espécies, conhecida como Computação Evolucionária (CE) (EIBEN; SMITH, 2003), motivou o surgimento da Computação Evolucionária Interativa (CEI). Dentre as definições para CEI encontradas na literatura, Takagi (2001) apresenta uma restrita e outra mais ampla. A primeira diz que "CEI é a tecnologia que aplica CE para otimizar sistemas alvos baseado na avaliação subjetiva humana como valor de aptidão". Já a segunda definição afirma que "CEI é a tecnologia na qual a CE otimiza sistemas alvo tendo uma interface interativa humano-máquina".

A união dos princípios da Computação Evolucionária aos conceitos da Otimização Interativa é potencialmente promissora, contudo, nem sempre é uma tarefa trivial (SHACKELFORD; SIMONS, 2014). Por exemplo, sabendo que as meta-heurísticas evolucionárias trabalham com populações de indivíduos, definir uma estratégia eficiente para a apresentação das informações referentes às soluções seria é um trabalho complexo. Isso porque o usuário poderá ter sua capacidade cognitiva afetada caso haja uma grande quantidade de dados a serem por ele considerados. Nesse sentido, Shackelford e Simons (2014) definem (1) a redução do tamanho da população a ser avaliada, (2) a seleção de alguns indivíduos para exibição e (3) a execução de múltiplas gerações entre as interações com o usuário, como padrões para apresentação de soluções em técnicas de Otimização Interativa.

Um aspecto fundamental da CEI é o processo de *feedback* humano, ou seja, como o usuário expressa suas preferências. Aljawawdeh, Simons e Odeh (2015a) estabelecem que as

preferências podem ser de natureza implícita ou explicita. No primeiro caso, as preferências são humanamente difíceis de articular, ou seja, o usuário as têm, mas não consegue claramente definilas. Dessa forma, ainda que não seja possível detectar os motivos que levam o humano a fazer uma determinada escolha, o mesmo pode contribuir para o processo de busca informando suas preferências através da atribuição de notas ou por comparação entre soluções. Já as preferências explícitas, são aquelas que o usuário pode articulá-las de forma precisa e, de modo geral, seu impacto no processo de busca é previamente conhecido.

O momento de captura das preferências do usuário é outro fator característico das abordagens que consideram a opinião humana para resolver problemas de otimização. De acordo com Deb (2012) e Miettinen (2012), existem três momentos nos quais as preferências podem ser capturadas:

a) Método *a priori*, no qual os desejos do usuário são concedidos antes do início do processo de busca;
b) Método interativo ou *"human-in-the-loop"*, no qual as preferências são fornecidas ao longo da execução do algoritmo de busca;
c) Método *a posteriori* no qual a opinião humana é informada após a finalização do processo de otimização.

2.3 ALGORITMOS GENÉTICOS

Os Algoritmos Genéticos (AGs) são algoritmos de busca inspirados nos mecanismos de evolução de populações de seres vivos e seguem os princípios da teoria de seleção natural e sobrevivência dos mais aptos, do naturista inglês Charles Darwin (LACERDA; CARVALHO, 1999). Inicialmente propostos por Holland (1975), os AGs são classificados como pertencentes à Computação Evolucionária (EIBEN; SMITH, 2003).

Existem diversas variações dos Algoritmos Genéticos, muitas delas especialmente desenvolvidas para contextos de aplicações ou categoria de problemas específicos. Por isso, a seguir serão mostrados apenas os conceitos fundamentais do que pode ser considerada uma versão canônica de Algoritmo Genético (AG).

Um AG explora o espaço de busca de um problema através da manutenção e evolução de uma população de indivíduos ao longo de um certo período. Cada indivíduo possui uma estrutura cromossômica que representa uma solução candidata para o problema. A evolução consiste na ação de operadores genéticos (tipicamente cruzamento e mutação) sobre a população,

na expectativa de que haja um contínuo melhoramento dos indivíduos. A avaliação do quão bom, ou seja, do quão apto para resolver o problema é um indivíduo, é dada por uma função de avaliação também chamada de *fitness*. De uma forma geral, em cada iteração do algoritmo, denominada de geração, a seleção dos indivíduos que sofrerão a ação dos operadores é influenciada por suas aptidões, assim, os mais aptos tendem a propagar seus materiais genéticos para as gerações futuras. Todo esse processo de funcionamento dos AGs é resumido pelo Algoritmo 1.

É válido ressaltar que, o comportamento dos operadores genéticos e do AG como um todo, está intimamente ligado à representação de solução adotada. Diversas representações podem ser usadas, tais como cadeia de *bits*, vetor binário e representação real. Nas explicações a seguir será adotada a representação binária na qual cada gene do cromossomo é preenchido com 0 ou 1.

Algoritmo 1: Algoritmo Genético canônico

```
Entrada: Instância do Problema
Saída: Indivíduo com maior aptidão
início
    Cria população inicial;
    Calcula aptidão dos indivíduos da população;
    enquanto não atingir o critério de parada faça
        Faz a seleção dos pais;
        Faz o cruzamento dos pais;
        Faz a mutação dos filhos;
        Calcula aptidão dos indivíduos da população;
        Atualiza população com novos filhos;
    fim
    retorna indivíduo com maior aptidão da população;
fim
```

O primeiro passo do AG é a criação da população inicial que tipicamente é realizada de forma aleatória. Tal estratégia é justificada pela expectativa da geração de soluções em diversas áreas do espaço de busca. Contudo, não são raras técnicas mais elaboradas para gerar populações iniciais de modo a privilegiar, por exemplo, áreas promissoras do espaço de soluções.

Após a geração dos primeiros indivíduos, é realizado o cálculo da aptidão de todos os indivíduos da população. A implementação desse procedimento é condicionada às características

do problema a ser resolvido e, por isso, é um aspecto fundamental para o funcionamento correto do processo de busca.

Depois da avaliação da primeira população, dar-se início ao processo evolucionário, o qual acontece iterativamente até que um certo critério de parada seja atingido. Os critérios para finalização da busca podem ser os mais variados, sendo a quantidade de gerações, limite de avaliações, tempo de execução, e gerações sem melhoria da melhor solução, os mais utilizados.

O processo evolucionário consiste na geração de populações com indivíduos cada vez mais adaptados para resolver o problema abordado. Para isso, o método de seleção dos pais que gerarão os indivíduos das próximas gerações é baseado no processo de seleção natural dos seres vivos observado por Darwin e Bynum (2009), ou seja, os seres mais adaptados tendem a sobreviver. Em outras palavras, os indivíduos com maior aptidão têm maior probabilidade de serem selecionados como pais e propagarem suas boas características para gerações futuras. Na prática, geralmente os pais são selecionados considerando-se uma probabilidade proporcional à sua aptidão, por exemplo, a probabilidade de seleção de um indivíduo p_i pode ser dada por

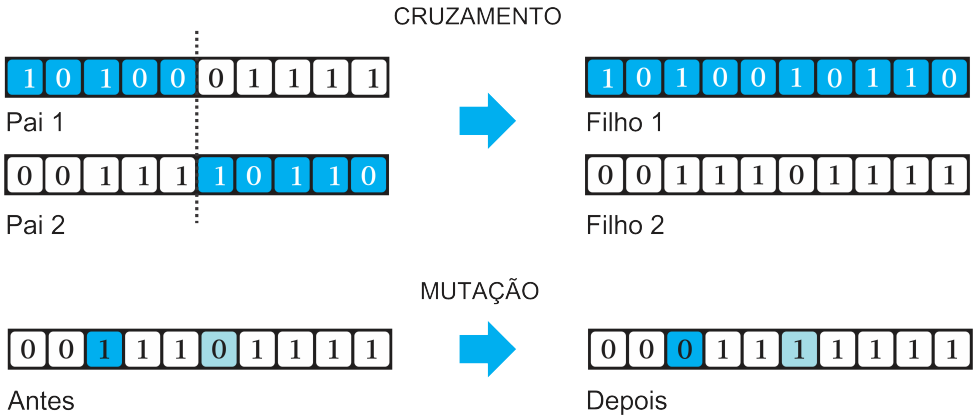
$$p_i = \frac{f_i}{\sum_{i=1}^{N} f_i}$$

onde N seria a quantidade de pais candidatos e f_i o *fitness* de cada indivíduo. O procedimento de seleção conhecido como roleta faz uso desse preceito. Outra técnica de seleção bastante utilizada é o torneio, cujo princípio consiste no sorteio de N indivíduos sendo o melhor deles selecionado como pai. Em ambas as técnicas a seleção dos pais prossegue até que a quantidade de pais desejada seja atingida.

Após o processo de seleção dos pais, é realizada a manipulação do material genético através dos operadores de cruzamento e mutação. Estes mecanismos desempenham funções distintas no que diz respeito á exploração do espaço de busca, porém, de forma objetiva, ambos atuam sobre a configuração do cromossomo. A Figura 2 mostra um exemplo de cada um desses operadores aplicados em indivíduos com cromossomos definidos por representação binária de solução.

O primeiro operador aplicado é o de cruzamento, que consiste em recombinar dois cromossomos pais gerando como resultado dois cromossomos filhos. Existem diversos tipos de cruzamento e, no caso das representações binárias, podem ser distinguidos pela quantidade de pontos de corte adotada. Ponto de corte é uma posição, aleatoriamente definida, que divide o par de cromossomos a serem cruzados. Quando se utiliza um ponto, a primeira parte dos dois pais é chamada de cauda e segunda de cabeça, assim, o filho 1 é constituído da cauda do pai 1

Figura 2 – Exemplos de Cruzamento e Mutação para uma representação binária



Fonte: Adaptado de Maia (2011).

com a cabeça do pai 2; por sua vez, o filho 2 recebe a cauda do pai 2 e a cabeça do pai 1. Ao se utilizar mais de um ponto de corte, os seguimentos gerados pelos cortes são alternadamente selecionados entre um pai e outro para constituir os filhos.

Na natureza, o fato de um par de indivíduos serem potenciais pais não determina que estes gerarão filhos. Nos AGs essa situação é simulada através da taxa de cruzamento, que representa a probabilidade de efetivamente ocorrer o cruzamento entre dois pais selecionados para tal. Geralmente a taxa utilizada varia entre 60% e 90% (SRINIVAS; PATNAIK, 1994).

Após a geração dos filhos, é realizado o processo de mutação o qual, analogamente ao que se verifica na natureza, proporciona a inclusão de novas informações nos cromossomos. Dessa forma, os filhos têm a possibilidade de receber características que não estejam presentes em seus pais. Conceitualmente, a mutação consiste numa alteração pontual e aleatória na configuração do cromossomo. Considerando uma representação binária, por exemplo, um método de mutação bastante difundido é o "*bit flip*", no qual percorre-se cada gene do cromossomo com uma probabilidade de $\frac{1}{N}$ de inverter o *bit*, sendo N a quantidade de genes do cromossomo (LINDEN, 2012).

Em termos de exploração do espaço de busca, os operadores de cruzamento e mutação possuem função complementar. Enquanto o primeiro possui características de busca local, ou seja, intensifica a busca em regiões próximas às das soluções representadas pelos indivíduos selecionados como pais, a segunda, possibilita a descoberta de soluções em áreas diferentes daquelas onde a busca se concentra num dado momento.

É imprescindível destacar que a maioria dos problema de otimização abordados com a aplicação de Algoritmos Genéticos, possui uma ou mais restrições que delimitam as

soluções válidas do espaço de busca. Por isso, mesmo assumindo a existência de um mecanismo que garanta a construção da população inicial de um AG contendo apenas soluções válidas, as recombinação genéticas nos cromossomos podem produzir indivíduos inválidos. A despeito dos conceitos biológicos, uma medida algorítmica que pode ser eficiente nesses casos é a criação de um procedimento de reparo que manipule a solução até que esta passe a respeitar todas as restrições do problema. Outras medidas podem ser a penalização da aptidão dos indivíduos inválidos ou mesmo o descarte destes.

Devido a natureza estocástica das manipulações no material genético, provocadas pelos operadores de cruzamento e mutação, existe a possibilidade de perda de informações relevantes com o passar das gerações. Nesse sentido, Jong (1975) propôs o mecanismo denominado elitismo. A ideia consiste na transferências dos *n* melhores indivíduos de uma geração para sua subsequente, garantindo assim que o indivíduo mais adaptado de uma geração nunca será pior do que o melhor da geração anterior.

Finalmente, a solução retornada pelo AG para um problema de otimização é sempre o indivíduo da última população, o qual contenha o melhor (maior ou menor dependendo do objetivo) valor de *fitness*, em outras palavras o indivíduo mais apto.

2.4 ENGENHARIA DE SOFTWARE BASEADA EM BUSCA

Assim como outras disciplinas de engenharia, a Engenharia de Software, possui inúmeros problemas relacionados aos métodos e processos que envolvem a construção e a evolução do seu produto alvo. Muitas atividades da ES são caracterizadas por conter restrições concorrentes, informações inconsistentes, grande quantidade de soluções candidatas ou até mesmo dificuldade de se definir precisamente o cômputo da qualidade de uma solução. Por isso, frequentemente, soluções perfeitas são impossíveis ou impraticáveis e a natureza de tais problemas pode dificultar a definição de algoritmos para resolvê-los (HARMAN; JONES, 2001).

Não obstante, atividades complexas como geração de casos de testes, refatoração, seleção de requisitos, alocação de tarefas, estimativas de custo e outras, podem ser formuladas como problemas de otimização e resolvidas por técnicas de busca, especialmente meta-heurísticas.

A subárea da Engenharia de Software dedicada a reformular seus problemas como problemas de otimização e resolvê-los com algoritmos de busca, é conhecida como Engenharia de Software Baseada em Busca, do inglês *Search-based Software Engineering* (SBSE). O termo SBSE foi introduzido por Harman e Jones (2001) num estudo onde foram apresentados alguns

apontamentos sobre a aplicação de meta-heurísticas em problemas inerentes ao desenvolvimento de software. Dentre os pontos fundamentais, o referido trabalho destaca os seguinte elementos como condição para a reformulação de um problema tradicional da Engenharia de Software em um problema de busca:

a) **Representação da solução:** é a definição simbólica de uma solução para o problema abordado. Tal elemento pode ser um vetor binário, um grafo ou mesmo uma árvore. O mais importante é este ser definido de forma que a manipulação da solução não seja demasiadamente complexa;
b) **Função de avaliação:** essa função é baseada na representação da solução e é utilizada como quantificador da qualidade referente as soluções candidatas;
c) **Operadores:** são mecanismos utilizados para manipulação das soluções de modo que seja possível gerar uma solução a partir de outra já existente. Suas características dependem da técnica de busca aplicada, mas de uma forma geral, é indispensável a presença de pelo menos um operador capaz de gerar pequenas alterações (também chamadas de perturbações) nas soluções, a fim de explorar o espaço de busca.

Como a Engenharia de Software possui métricas utilizadas em muitas de suas atividade corriqueiras, a definição dos pré-requisitos mostrados anteriormente torna-se bastante factível, assim, uma vez definida a representação da solução, a função de avaliação e um método de transição entre soluções, a resolução do problema faz-se viável pela aplicação das técnicas de otimização já amplamente difundidas e consolidadas.

Contudo, o fato de uma atividade da ES poder ser modelada matematicamente como problema de otimização, não significa que se deve assim fazer, muito menos que uma métrica previamente existente seja a melhor escolha para guiar o processo de busca por soluções. Por esse motivo, Harman e Clark (2004) destacam aquilo que deve ser considerado para que as métricas próprias da ES (ou quaisquer outras) possam ser empregadas como função objetivo e, consequentemente, sejam obtidos, por intermédio de uma técnica de busca, resultados realísticos e relevantes. Os aspectos requeridos são:

a) **Espaço de busca com grande extensão**: por si só, a existência de uma métrica capaz de distinguir qualitativamente soluções candidatas não justifica o uso de SBSE, pois se o espaço de busca for pequeno, uma busca exaustiva é suficiente;
b) **Baixa complexidade computacional para avaliar soluções**: normalmente as

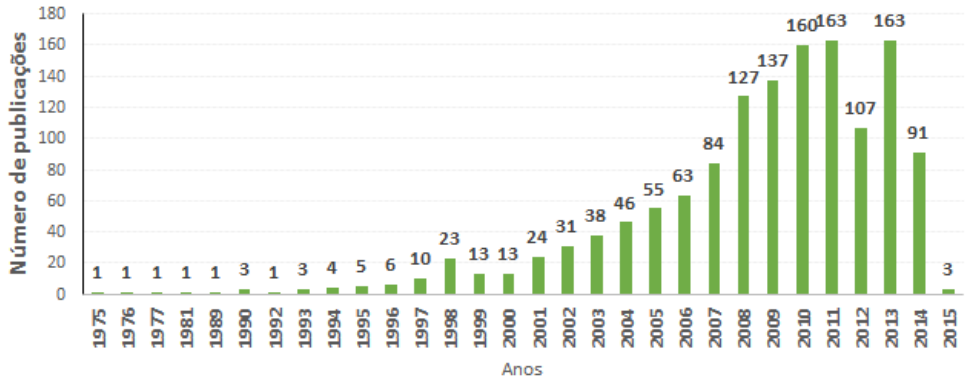
técnicas de busca “varrem” diversas áreas do espaço possível de soluções o que naturalmente requer elevado número de avaliação ao longo de todo o processo. Dessa forma, dependendo da complexidade computacional para se calcular a métrica utilizada como função de avaliação, pode ser inviável esperar todo o tempo necessário para se conseguir uma boa solução;

c) **Continuidade aproximada**: se a função de avaliação for descontínua o algoritmo de busca pode não conseguir informações suficientes para guiá-lo corretamente pelo espaço de busca e por consequência perder a capacidade de encontrar áreas promissores;

d) **Ausência de soluções ótimas conhecidas**: obviamente, se já existir algum método com a capacidade encontrar a solução ótima não faz sentido utilizar uma meta-heurística que não oferece tal garantia.

A necessidade de formalização e automatização de atividade da Engenharia de Software, aliadas à ampla consolidação do conhecimento na área da Otimização, têm feito da Engenharia de Software Baseada em Busca um promissor campo de pesquisa. A Figura 3 mostra um levantamento das publicações por ano em SBSE. Percebe-se que apesar do primeiro trabalho caracterizado dentro dessa subárea ser datado de 1975, a comunidade tornou-se bem mais ativa nas duas últimas décadas, chegando hoje a mais de 1300 [1] produções científicas publicadas em eventos específicos de SBSE ou ES e de computação em geral.

Figura 3 – Publicações por ano em SBSE



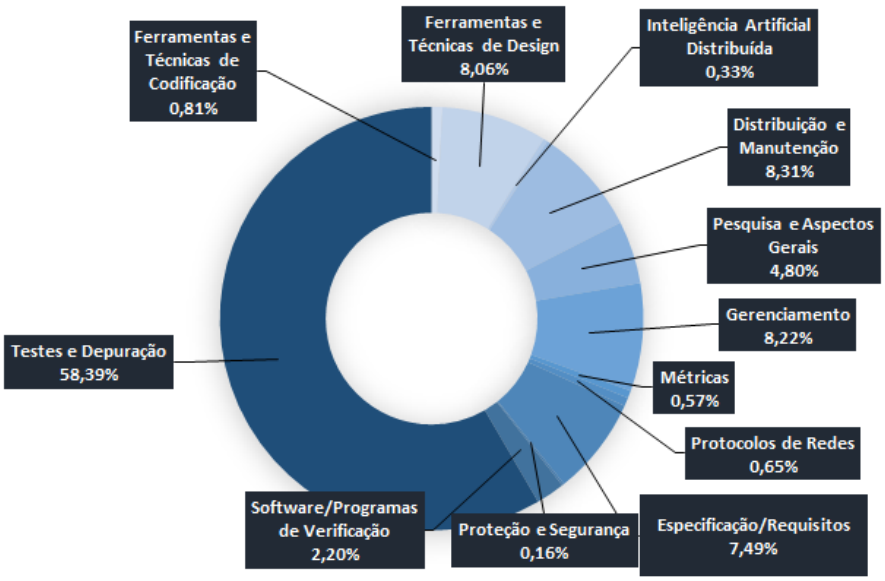
Fonte: Adaptado de (ZHANG; MANSOURI, 2015).

As aplicações de SBSE têm alcançado diversos campos da Engenharia de Software,

[1] A quantidade exata é 1389, obtida de <http://crestweb.cs.ucl.ac.uk/resources/sbse_repository/repository.html> em 05/08/2015

uma prova disto são, por exemplo, os trabalhos de Kukunas, Cupper e Kapfhammer (2010), Jiang *et al.* (2007) e Hart e Shepperd (2002) no contexto de ferramentas e técnicas de codificação; as pesquisas de Guizzo, Colanzi e Vergilio (2014), Simons, Smith e White (2014) e Ramrez, Romero e Ventura (2013) em ferramentas e técnicas de design; na área de testes e depuração é possível citar as contribuições de Assunção *et al.* (2014), Ali e Iqbal (2014) e Aburas e Groce (2014). Ademais, a Figura 4 mostra a distribuição dos trabalhos de SBSE em 11 área de aplicação catalogadas por Zhang e Mansouri (2015).

Figura 4 – Taxa de publicações em SBSE por área de aplicação



Fonte: Adaptado de (ZHANG; MANSOURI, 2015).

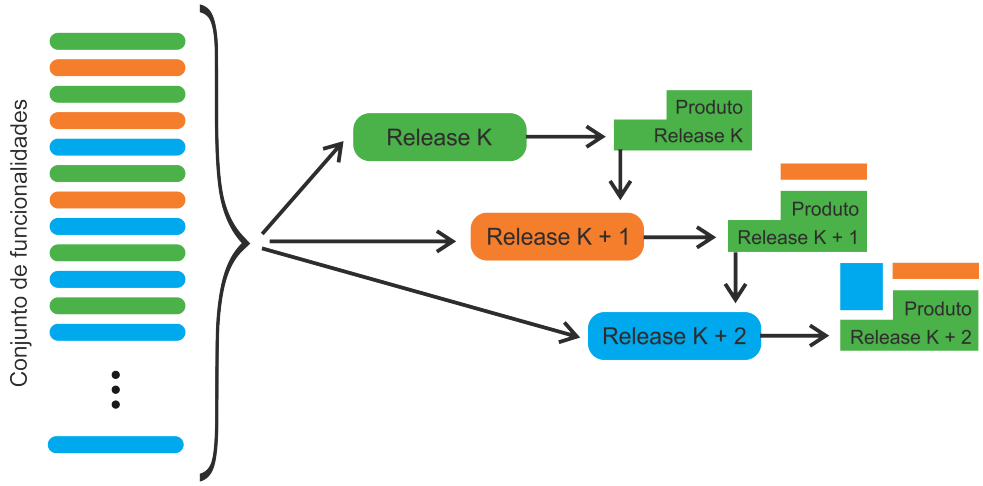
Como revelam os dados, mais da metade das pesquisas em SBSE são realizadas na área de testes e depuração, inclusive todos os trabalhos até a década de 90 foram desenvolvidos em tal contexto. A ampla exploração e o forte apelo à automatização de processos relativos a testes, são aspectos que naturalmente instigam o uso das técnicas de otimização nesse escopo da Engenharia de Software. Entretanto, outras áreas como ferramentas e técnicas de design, gerenciamento, distribuição e manutenção e especificação/requisitos têm sido significativamente abordadas e juntas somam mais de 30% dos 1228 trabalhos classificados por área de aplicação.

2.5 PLANEJAMENTO DE *RELEASES*

O Planejamento de *Releases* (PR) é, em essência, o processo de definir as funcionalidades de uma sequência de versões de um produto desenvolvido de forma incremental

(RUHE, 2010). Para tanto, deve ser empregado um esforço na definição de quais requisitos serão implementados em cada entrega, de modo a contemplar as restrições técnicas, de orçamento, de recursos e de riscos do projeto (RUHE; SALIU, 2005).

Figura 5 – Ilustração do Planejamento de *Releases*



Fonte: Elaborada pelo autor.

A Figura 5 representa o planejamento das *releases* de um software, onde cada versão do produto é um incremento da versão anterior mais as funcionalidade selecionadas para desenvolvimento na versão atual. Entretanto, a definição de quais requisitos a serem implementados em cada versão deve ser realizada considerando-se diversos aspectos técnicos e organizacionais, os quais serão apresentados a seguir.

2.5.1 Principais aspectos do Planejamento de *Releases*

Como o PR é reconhecidamente uma atividade desafiadora, a seguir serão apresentados alguns aspectos para ajudar na compreensão do quão complexa é essa atividade e nortear a aplicação de estratégias para tratamento do problema. Segundo Saliu e Ruhe (2005) os principais fatores que caracterizam o Planejamento de *Releases* são:

a) **Escopo**: diz respeito a quantidade de *releases* que se deseja planejar. Várias abordagens como as propostas por Baker *et al.* (2006), Ferreira e Souza (2012) e Fuchshuber e Barros (2014) proporcionam o planejamento de apenas uma *release*, o que pode não ser suficiente, pois os clientes envolvidos podem ficar desapontados ao não verem suas prioridades atendidas e não terem uma previsão de quando serão. Por outro lado, considerando a natureza dinâmica do problema,

projetar muitas *releases* pode não fazer muito sentido;

b) **Horizonte de tempo**: refere-se ao intervalo de tempo que o produto ou parte dele é liberado, ou seja, é o ciclo de entrega. Basicamente, os planejamentos podem ter os intervalos de tempo para desenvolvimento fixo e pré-definidos ou flexíveis;

c) **Objetivos**: define o que se pretende alcançar com o planejamento. Tipicamente, as técnicas para planejamento de *releases* consideram vários aspectos para definição dos objetivos. Os mais comuns são: valor de negócio, urgência das funcionalidades, risco do projeto, satisfação dos *stakeholders* e retorno de investimento;

d) ***Stakeholder***: representa uma pessoa ou organização que possui algum interesse no projeto de desenvolvimento do software. O envolvimento dos *stakehoders*, sejam eles desenvolvedores, clientes, testadores ou gerentes de projeto, durante o planejamento é essencial para a qualidade final do produto, porém na maioria das vezes eles acabam não se envolvendo o suficiente;

e) **Mecanismo de priorização**: é a identificação do quão prioritária é uma funcionalidade com relação a todas as outras que também precisam ser implementadas. De uma forma geral a prioridade é estabelecida por sistema de votos dados pelos *stakeholders*, sem considerar esforço, risco ou outras demandas ligadas à implementação;

f) **Restrições técnicas**: são interações ou relações que as funcionalidades possuem entre si. Por exemplo, restrições técnicas podem se referir a dependência de acoplamento, na qual duas ou mais funcionalidades devem ser implementadas juntas; ou dependência de precedência quando uma funcionalidade X deve ser implementada antes de uma funcionalidade Y e vice-versa. O estudo de Carlshamre *et al.* (2001) sobre um repositório de requisitos do domínio de telecomunicações revelou que cerca de 20% dos requisitos eram independentes, evidenciando assim, a relevância de se considerar as dependências entre requisitos;

g) **Restrições de recursos**: o tomador de decisão precisa considerar várias restrições de recursos durante a alocação de funcionalidades ou requisitos em *releases*. Geralmente, essas restrições dizem respeito a diferentes tipos de recursos, cronogramas, orçamentos, riscos e esforços inerentes à todas as etapas de

desenvolvimento;

h) **Restrições de sistema**: é basicamente a avaliação do efeito que as funcionalidades terão sobre sistemas já existentes. Podem ser utilizadas arquiteturas existentes, base de códigos, repositórios de mudanças e histórico de defeitos como fonte de informações para ajudar na quantificação de esforço extra, risco e complexidade de novas funcionalidades;

i) **Caráter e qualidade das soluções oferecidas**: refere-se àquilo que pode ser considerado e aceito como solução para o problema. Por exemplo, uma única solução *versus* um conjunto de soluções, uma solução sem qualidade definida *versus* uma solução que alcance um certo nível de qualidade pré-estabelecido;

j) **Ferramenta de suporte**: o Planejamento de *Releases* é um intenso processo com diversas tarefas humanamente exaustivas como estimativa de recursos, conflitos entre interesses de *stakeholders* e todos os aspectos discutidos anteriormente. Por tudo isso, ferramentas de suporte inteligentes que possam auxiliar o tomador de decisão no processo de decisão são um meio de agregar valor ao resultado final.

2.5.2 Planejamento de Releases como Problema de Busca

Tal como outros problemas da Engenharia de Software, o planejamento de *releases* apresenta algumas características de uma atividade que pode ser modelada matematicamente como problema de busca e resolvido de forma automatizada. Com esse intuito, Bagnall, Rayward-Smith e Whittley (2001) pioneiramente presentaram uma estratégia de resolução do Planejamento de *Releases* como problema de Otimização Matemática. A técnica, no entanto, tratou o planejamento de uma única *release*, o que ficou caracterizado como um tipo especial de planejamento, denominado Problema do Próximo *Release*, do inglês *Next Release Problem* (NRP).

Posteriormente, Ruhe e Saliu (2005) apresentaram uma formulação mais flexível com relação ao número de *releases*. Além disso, vários fatores intrínsecos ao problema foram adicionados ao modelo matemático, tornando-o ainda mais realista. A referida abordagem é uma das mais citadas na literatura desse contexto e por isso será utilizada a seguir para exemplificação de como o Planejamento de *Releases* pode ser abordado numa perspectiva de problema de busca.

Supondo um conjunto de funcionalidades $F = \{f(1), f(2), \cdots, f(n)\}$ no qual podem conter novas funcionalidades, alterações requeridas por clientes ou correção de defeitos, o objetivo é alocar todos os itens de F num conjunto finito de K *releases*. Isto pode ser representado

pelas variáveis de decisão $\{x(1), x(2), \cdots, x(n)\}$, onde $x(i) = k$ se o requisito i for alocado na *release* $k \in \{1, 2, 3 \cdots, K\}$ e $x(0)$ caso contrário.

Dentre as dependências que podem ocorrer entre os requisitos, a formulação considera as de Acoplamento e as de Precedência. Funcionalidades com acoplamento entre si devem ser implementados numa mesma *release* porque cada uma depende da outra. A Precedência indica uma certa ordem de implementação dos requisitos quando for impossível o desenvolvimento de alguma funcionalidade sem que antes uma outra esteja disponível. Tais restrições podem ser expressadas como:

$$x(i) = x(j) \forall (i, j) \in C \text{ (Acoplamento)},$$

$$x(i) \leq x(j) \forall (i, j) \in P \text{ (Precedência)},$$

onde C é o conjunto de todos os pares de requisitos com a relação de acoplamento entre si, e, P o conjunto com as relações de precedência entre requisitos.

Para tratamento das restrições de recursos, é considerado a capacidade máxima disponível em cada recurso. Assumindo T tipos de recursos, $Cap(k,t)$ é a capacidade que cada *release* $k = 1...K$ possui para cada tipo de recurso $t = 1...T$. Toda funcionalidade $f(i)$ requer para seu desenvolvimento uma quantidade $r(i,t)$ de recursos do tipo t. Assim, a atribuição dos requisitos para as *releases* devem satisfazer:

$$\sum_{x(i)=k} r(i,t) \leq Cap(k,t),$$

para todo *release* k e recursos t.

Os *stakeholders* são extremamente importantes para o planejamento de um produto, porém, nem todos possuem o mesmo nível de importância dentro de processo de desenvolvimento. Por isso, assumindo $S = \{s(1), \cdots, s(q)\}$ como o conjunto de todos *stakeholders* envolvidos, faz sentido atribuir uma importância relativa $\lambda(p) \in \{1, 2, 3 \cdots, 9\}$ para cada um deles. Este intervalo de valores é utilizado para permitir uma suficiente distinção entre a relevância dos *stakeholders*.

Para priorização das funcionalidades, muitos critérios podem ser utilizados. Na formulação proposta por Ruhe e Saliu (2005) foram considerados o valor de negócio e urgência da implementação dos requisitos, ambos expressos por notas do intervalo [1,9]. Cada *stakeholder*

define um valor $value(p,i) \in \{1,2,\cdots,9\}$ representando o quão importante é para ele a funcionalidade i, de modo que $value(p,i) = 1$ e $value(p,i) = 9$ representariam a menor e a maior relevância, respectivamente. A urgência é uma forma de modelar a preferência dos *stakeholders* pela implementação de uma dada funcionalidade numa certa *release*. Assumindo o planejamento de duas *releases*, cada *stakeholder* teria 9 votos para cada funcionalidade e deveria distribuí-los entre 3 opções de alocação: *release* 1, *release* 2 ou não ser alocado. Quanto mais votos receber uma opção, maior será a satisfação do *stakeholder* se no planejamento final aquela funcionalidade for alocada na referida opção. Assim, uma urgência $urgency(p,i) = (9,0,0)$ indicaria que o *stakeholder* $S(p)$ tem máximo desejo que o requisito i seja alocado na *release* 1.

Conforme já mencionando, os objetivos do planejamento de *release* podem ser variados. Na pesquisa de Ruhe e Saliu (2005) os objetivos referentes ao valor e a urgência foram compostos e ponderados numa única função $F(x)$, assim uma técnica de otimização deveria maximizar $F(x)$ respeitando todas as restrições consideradas no problema. Formalmente, esta função é defina como

$$F(x) = \sum_{k=1\cdots K} \sum_{i:x(i)=k} WAS(i,k),$$

onde:

$$WAS(i,k) = \xi(k) \left[\sum_{p=1,\cdots,q} \lambda(p) \times value(p,i) \times urgency(p,i,k) \right], \tag{2.1}$$

sendo $\xi(k)$ a importância relativa de cada *release* dada por um valor normalizado.

Ainda falando sobre objetivos, outros fatores, como por exemplo o risco do projeto, são comumente considerados na literatura como pode ser constatado em Souza *et al.* (2011), Colares *et al.* (2009) e Brasil *et al.* (2011) sendo que, nestas duas últimas, as abordagens propostas fazem uso de otimização multiobjetivo.

2.6 ANÁLISE ESTATÍSTICA DE META-HEURÍSTICAS EM *SBSE*

As meta-heurísticas, que são as principais técnicas de busca utilizadas em SBSE, possuem caráter estocástico ou aleatório e, por esse motivo, a avaliação comparativa dos resultados por elas obtidos é uma tarefa que exige cuidado especial. Como execuções diferentes desses algoritmos tendem a produzir valores distintos, é imprescindível o uso de análises estatísticas para garantir maior confiabilidade às conclusões sobre os resultados atingidos.

De acordo com o levantamento feito por Barros e Dias-Neto (2011b), muitas das avaliações de resultados de meta-heurística aplicadas em Engenharia de Software, são baseadas em elementos da Estatística Descritiva como média, moda, desvio padrão, etc. Tipicamente esses dados são coletados a partir de um número n de execuções do algoritmo ou configuração sob análise. Entretanto, Arcuri e Briand (2011) discutem que para este contexto, considerar apenas os valores de média, por exemplo, pode levar à conclusões equivocadas dada a grande variância da Probabilidade de Distribuição. Por isso, os mesmo autores recomendam a utilização de testes estatísticos para a avaliação de resultados produzidos por meta-heurística em SBSE.

Um teste estatístico é empregado quando se pretende, por exemplo, verificar se duas amostras são estatisticamente diferentes, ou seja, se há diferença empírica entre os resultados produzidos por duas técnicas ou configurações distintas. Nesses casos, geralmente assume-se uma hipótese nula H_0 de que as amostras não são diferentes. Por consequência, duas situações podem ocorrer: a primeira, chamada erro 1 é quando se rejeita a hipótese nula quando na verdade ela é verdadeira. A segunda, dita erro 2, é quando aceita-se a hipótese sendo ela falsa. Com isso, os testes encontram um *p-value* que é a probabilidade de ocorrência do erro 1 na comparação entre as amostras. Outro importante elemento desse tipo de teste é o nível de significância α, o qual representa o maior valor de *p-value* admitido para rejeitar a hipótese nula (ARCURI; BRIAND, 2011).

Para a análise de diferença estatística, dois tipos de testes podem ser aplicados de acordo com a distribuição probabilística apresentada pelos dados das amostras. No caso das amostras seguirem uma distribuição normal, devem ser utilizados testes paramétricos como o *Student T*, caso contrário, recomenda-se o uso de testes não-paramétricos como por exemplo, teste *U de Mann-Whitney* (ARCURI; BRIAND, 2011).

Apesar de bastante utilizados para demonstração de diferença estatística entre amostras, os testes de significância por hipóteses nulas não oferecem informações sobre a magnitude do evento observado, tampouco conferem precisão de estimativa estatística do efeito da magnitude (NAKAGAWA; CUTHILL, 2007). Por esse motivo, em algumas ocasiões é relevante o uso de testes estatísticos denominados como *effect size*, ou magnitude de efeito. Num contexto de observação dos resultados provenientes de dois algoritmos A e B, um *effect size* pode ser usado para determinar o quão mais eficiente é um deles.

Um teste de magnitude de efeito comumente verificado na literatura é o *Varga and Delaney A*. Nesse teste, a estimativa estatística é a probabilidade de que um valor aleatoriamente

selecionado da amostra 1 seja numericamente maior do que um outro valor também aleatoriamente selecionado da amostra 2, tal comparação é representada por $\hat{A}_{12}$. Vargha e Delaney (2000) estabelecem que a magnitude é **pequena** se a estimativa for maior que $0,56$ e menor que $0,64$, **média** se $0,64$ e menor que $0,71$, e por fim, **grande** entre $0,71$ e 1.

2.7 CONCLUSÕES DO CAPÍTULO

Neste Capítulo foram apresentados os conhecimento que embasam o presente estudo. Inicialmente foi esclarecido que a otimização matemática ocupa-se em modelar diversos problemas do mundo real e resolvê-los a partir do emprego de critérios matemáticos que qualifiquem possíveis soluções. Adicionalmente, foi mostrada a Otimização Interativa como tipo de otimização matemática na qual busca-se integrar o conhecimento humana à capacidade computacional dos algoritmos de otimização.

Como exemplo de algoritmo de busca bastante utilizado para resolução de problemas de otimização, foram apresentados os Algoritmos Genéticos (AGs), sendo explicado que estes possuem inspiração biológica pois o funcionamento dos mesmos simula o processo de seleção natural verificado na natureza.

Foi apresentada uma área de pesquisa denominada Engenharia de Software Baseada em Busca, ou *Search-based Software Engineering* (SBSE), a qual destina-se a tratar questões complexas da Engenharia de Software como problemas de otimização. Viu-se que esta é, apesar de recente, uma área que tem despontado como promissora dada a quantidade de publicações e a abrangência de contextos alcançados dentro da ES.

A atividade de planejar as *releases* de um software foi apresentada como uma tarefa complexa, que possui diversos aspectos a serem considerados pelo profissional responsável por esta atividade. Por esse motivo, a referida tarefa tem sido modelada como problema de busca possível de resolução por técnicas de otimização.

Finalmente, foi destacado que a análise estatística em experimentos no contexto de SBSE é um fator preponderante para a consistência das conclusões obtidas. Por esse motivo, além múltiplas execuções, foi também aconselhado emprego de teste que verifiquem a diferença estatística entre amostras e teste que mensure a magnitude de efeito.

3 TRABALHOS RELACIONADOS

Neste capítulo serão apontados alguns trabalhos que se relacionam com os aspectos fundamentais da presente pesquisa. Inicialmente serão apresentadas estudos que abordam o Planejamento de *Releases* como problema de otimização, considerando apenas pesquisas focadas no planejamento de uma sequência de *releases*, ou seja, excetuando-se o NRP. Em seguida serão discutidas pesquisas no contexto da Engenharia de Requisitos, as quais tenham aplicado Otimização Interativa. E, por fim, serão mostrados exemplos de técnicas que exploram Computação Evolucionária Interativa fazendo uso de preferências explícitas.

3.1 PLANEJAMENTO DE *RELEASES* EM SBSE

Ruhe e Ngo-The (2004) apresentaram a proposta de uma abordagem para planejamento de *releases* chamada de EVOLVE*, a qual combina a capacidade humana para modelagem matemática do referido problema, com o potencial da inteligência computacional em gerar boas soluções. A abordagem proposta divide-se em 3 fases: Modelagem, Exploração e Consolidação. Na primeira, define-se a modelagem do problema, ou seja, restrições, objetivos, estimativas e outros. Na segunda, utiliza-se um algoritmo genético para encontrar soluções candidatas e por fim são realizados, quando necessário, ajustes na modelagem. Adicionalmente, Amandeep, Ruhe e Stanford (2004) e Greer e Ruhe (2004) reforçaram a aplicabilidade de estratégias de planejamento apoiadas pelo EVOLVE*.

Posteriormente surgiram outras pesquisas fundamentadas nos mesmos princípios, as quais basicamente apresentam alguma extensão das anteriores. Por exemplo, na pesquisa de Ruhe e Saliu (2005), a fase 2 (exploração), é complementada com utilização de programação inteira para promover a geração de um conjunto de soluções alternativas. Já Ngo-The e Ruhe (2008) adicionaram o conceito de restrições leves e utilizam um método chamado ELECTRE-IS (FIGUEIRA; MOUSSEAU; ROY, 2005) para escolher dentre um conjunto de soluções candidatas, aquela que melhor atendesse tais restrições.

Sabendo da eficiência dos algoritmos baseados em Colônia de Formiga aplicados a vários problemas de otimização combinatória, Souza *et al.* (2011) propuseram a primeira adaptação de um ACO para o Planejamento de *Releases*. Para isso, apresentaram uma formulação mono-objetivo visando maximizar o valor de negócio e minimizar o risco do projeto, considerando as restrições de custo de cada *release* e as precedências técnicas entre requisitos.

O ACO foi composto por dois tipos de arestas, um para representar os movimentos opcionais das formigas e outro definindo os movimentos obrigatórios, simulando assim, a restrição de precedência. Além disso, em cada iteração do (ACO), cada formiga partindo de um vértice aleatório selecionava apenas vértices (requisitos) de uma única *release k*, após todas as formigas terminarem seu percurso, o processo era repetido para a *release* $k+1$ até formar uma solução. Os resultados obtidos a partir de 72 instâncias artificiais mostraram que o ACO foi superior às meta-heurísticas AG e Têmpera Simulada em termos de *fitness*, porém apresentou alto consumo de tempo quando requeridas muitas avaliações.

Em razão do Planejamento de *Releases* possuir inúmeros fatores e estes serem muitas vezes conflitantes, Colares *et al.* (2009) e Brasil *et al.* (2011) apresentaram abordagens com formulações multiobjetivo para o referido problema. Ambas as propostas consideram fatores como a prioridade, o risco do projeto e as interdependências entre os requisitos. Porém, enquanto o primeiro trabalho visa maximizar a satisfação dos clientes e minimizar o risco do projeto em um número fixo de *releases*, o segundo propõe uma formulação para número indefinido de *releases* e outra para uma quantidade desejável de *releases*.

No trabalho de Colares *et al.* (2009), a alocação dos requisitos maximiza a satisfação e minimiza o risco do projeto através da implementação antecipada dos requisitos com maior prioridade e maior risco, respectivamente. Ao tempo em que são respeitadas as restrições de orçamento e precedência técnica.

Já na pesquisa de Brasil *et al.* (2011), há inicialmente uma fase de seleção e priorização dos requisitos na qual busca-se maximizar o valor de negócio, minimizar o risco do projeto e maximizar o valor da prioridade de implementação das funcionalidades, considerando as restrições de tempo, orçamento e as interdependências de acoplamento, precedência técnica e de negócio. A partir de então, ocorre a alocação dos requisitos, objetivando minimizar o acoplamento entre as *releases*, minimizar a quebra das precedências de negócio e tentando manter a priorização obtida na fase anterior.

Como se percebe, o planejamento de *releases* é um problema vastamente abordado em SBSE, entretanto, diferente do método proposto neste trabalho nenhuma das pesquisas acima relacionadas possuiu foco na modelagem da opinião do tomador de decisão, e por conseguinte não o envolviam de maneira efetiva no processo de geração de soluções.

3.2 OTIMIZAÇÃO INTERATIVA EM ENGENHARIA DE REQUISITOS

Com relação à aplicação de otimização interativa em problemas da Engenharia de Requisitos, Tonella, Susi e Palma (2010) apresentam uma abordagem interativa para priorização de requisitos. Nela, o objetivo é estabelecer uma ordem de implementação dos requisitos minimizando a divergência entre a prioridade e as relações de dependência entre os requisitos. A interatividade com o usuário foi utilizada para fornecer informações adicionais ao Algoritmo Genético Interativo (AGI) quando este se deparava com indivíduos igualmente aptos. Nesses casos eram identificados todos os pares de requisitos com ordens diferentes nas duas soluções e o usuário informava qual deveria ser a ordem de precedência entre ambos. A resposta era então inserida num grafo de elicitação de precedências que passava a ser também considerado pelo AGI durante a avaliação da população. Foi demonstrado que o AGI supera um Algoritmo Genético convencional com relação a qualidade da priorização final produzida por cada técnica.

Enquanto no trabalho anterior a interação com o usuário é requerida apenas em alguns momentos chave, Araújo e Paixão (2014) apresentaram um trabalho onde a interatividade é utilizada na avaliação de todas as soluções de um AGI aplicado ao NRP. O trabalho deles propõe um modelo de otimização mono-objetivo que adapta a formulação introduzida por Baker *et al.* (2006) adicionando à função de avaliação das soluções, um componente responsável por encapsular uma avaliação humana para cada solução. Nesse trabalho, o processo de geração de soluções é realizado por um AGI, sendo cada indivíduo avaliado pelo usuário através de uma nota que adicionada aos critérios próprios do NRP compõem a função de aptidão dos indivíduos. Através da contabilização das preferências geradas por um simulador do tomador de decisão, demonstrou-se que as soluções geradas pela abordagem proposta de fato foram influenciadas pela opinião humana.

Diferente das duas pesquisas anteriores que utilizaram algoritmos genéticos, Ferreira (2015) apresentou um estudo sobre formas de incorporar interatividade na resolução do NRP através de Otimização por Colônia de Formigas. Dentre as 7 estratégias de interatividade propostas, o autor implementou e avaliou apenas uma delas. A estratégia escolhida, concede ao usuário a possibilidade de informar sua opinião individualmente para cada requisito, expressando se gostaria de vê-lo no próximo *release*, se gostaria de não vê-lo no próximo *release* ou ainda, indicar que não possui preferência sobre o requisito. As predileções do usuário são armazenadas e utilizadas para balancear a probabilidade de escolha de um requisito durante o processo construtivo das soluções pelo algoritmo de busca. Também considerando a métrica de percentual

de preferências satisfeitas, os experimentos demonstraram que a técnica interativa atendeu em média cerca de 80% contra aproximadamente 50% de uma versão não interativa do mesmo algoritmo.

Recentemente, Pitangueira (2015) apresentou uma proposta de pesquisa para incorporação de preferências de múltiplos *stakeholders* durante o processo de seleção de requisitos. A proposta pretende utilizar otimização interativa para explorar regiões do espaço de busca que sejam mais preferíveis para os usuários, contudo o autor ainda não definiu como será o processo de captura das preferências, a natureza destas (implícitas ou explícitas) e como serão utilizadas no processo de otimização. Segundo a proposta, serão considerados pelo menos três objetivos envolvendo valor de negócio, custo e risco de implementação e no momento estão sendo estudadas as técnicas de otimização que melhor se adequem ao problema.

Não obstante ao fato dos trabalhos apresentado nesta seção fornecerem consistente aplicabilidade das técnicas de otimização interativa à situações práticas da engenharia de requisitos, todos eles restringem-se ao cenário do planejamento de uma única *release* (NRP). Além disso, em nenhuma das pesquisas fora considerada a possibilidade diferentes preferências do tomador de decisão poderem possuir diferentes níveis de urgência.

3.3 CEI COM PREFERÊNCIAS EXPLICITAMENTE DEFINIDAS

Thiele *et al.* (2009) e mais recentemente Ruiz, Saborido e Luque (2014) apresentaram abordagens combinando as características de otimização interativa com técnicas evolucionárias para problemas multi-objetivos. Em ambas as propostas, para cada geração da população, o tomador de decisão tinha a responsabilidade de informar explicitamente a área da frente de Pareto sobre a qual o mesmo possuía mais preferência. A informação é então utilizada por um algoritmo evolucionário baseado em preferências, o qual gera a nova população combinando os valores da função objetivo convencional e de uma função de aproximação do ponto de referência obtido a partir da interação.

Já no trabalho de Duenas, Tütüncü e Chilcott (2009), foi aplicado um AG mono-objetivo em conjunto com uma técnica *fuzzy* para resolver o problema da escala de enfermeiros num ambiente hospitalar. Para isso, foram consideradas restrições fortes e leves (estas últimas eram as próprias preferências dos enfermeiros). Um sistema *fuzzy* foi utilizado para que a função de avaliação pudesse considerar adequadamente a quebra de restrições leves referentes a cada enfermeiro, sabendo que os profissionais são diferentemente afetados por uma mesma

solução e os tipos de preferências possuíam níveis diferentes de impacto ao serem violados. O tomador de decisão, ou seja, a pessoa responsável por definir a escala do enfermeiros, estabelecia suas preferências com relação a níveis de qualidade das soluções candidatas para cada aspecto considerado pela função de avaliação, e, em cada iteração tinha a possibilidade de alterá-las e continuar a execução do AG até que um critério de parada fosse atingido.

As referências anteriores reforçam o uso do conhecimento humano conjuntamente às técnicas de busca evolucionárias como forma de guiar o processo de busca por regiões do espaço que sejam de maior relevância do usuário.

3.4 CONCLUSÕES DO CAPÍTULO

Neste Capítulo foram apresentados trabalhos que se relacionam a algum dos aspectos mais importantes da presente pesquisa. Inicialmente, foi demonstrada uma linha de evolução das abordagens de SBSE para Planejamento de *Releases* ao longo do tempo, de modo geral, variadas técnicas de busca já foram aplicadas ao problema e formulações mono-objetivo e multiobjetivo foram concebidas.

Viu-se que no contexto da engenharia de requisitos, a otimização interativa vem sendo aplicada por abordagens focadas em priorização ou seleção de requisitos para o *Next Release Problem*. Os Algoritmos Genéticos Interativos foram utilizados tanto para acomodar preferências implícitas (avaliação subjetiva para uma solução do NRP) como explícitas (indicação de prioridade). Já a técnica ACO foi utilizada para solucionar requisitos considerando opiniões do tomador de decisão explicitamente definidas ao longo de processo de busca.

Por fim, foram apresentados exemplos de abordagens fora do contexto de SBSE, mas que reforçaram a aplicabilidade de conhecimento humano à técnicas de busca evolucionárias como forma de guiar o processo de busca por regiões do espaço que sejam de maior relevância para o tomador de decisão, inclusive considerando diferentes níveis de prioridade para as preferências.

Apesar do Planejamento de *Releases* ser um problema bastante explorado em SBSE, constata-se que ainda não existem abordagens focadas na modelagem das preferências do tomador de decisão a fim de incorporá-lo ao longo de processo de otimização.

4 ABORDAGEM PROPOSTA

Este capítulo apresenta os detalhes da abordagem que este trabalho propõe para o Planejamento de *Releases* (PR). Inicialmente é mostrada, na Seção 4.1, uma visão geral do funcionamento da técnica e dos elementos que a compõem. Em seguida, na Seção 4.2, é descrita a modelagem adotada para o PR, destacando-se os detalhes relativos aos requisitos, *releases*, clientes e representação da solução. Posteriormente a Seção 4.3 apresenta o levantamento e a definição formal das preferências do usuários, para que na sequência, em 4.4, seja explicada em detalhes a formulação matemática do referido problema. Finalmente, a Seção 4.5 traz um cenário fictício para exemplificar a aplicabilidade da abordagem proposta.

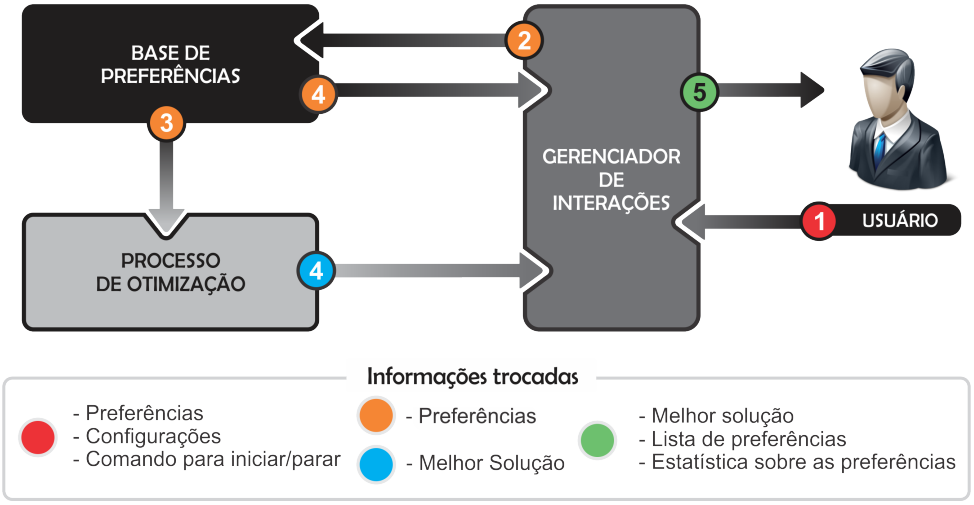
> Uma estratégia de planejamento de *releases* objetiva atribuir funcionalidades para *releases* tal que restrições de orçamento, de risco e técnicas sejam atendidas. Uma vez que uma estratégia de planejamento de *releases* tenha sido gerada, ou seja, tomada a decisão sobre quais funcionalidades devem ser desenvolvidas em cada *release*, a operacionalização do planejamento se concentra no desenvolvimento das funcionalidades identificadas numa única *release* do software. (AL-EMRAN; PFAHL, 2007)

A técnica de resolução do Planejamento de *Releases* proposta por este trabalho baseia-se na definição de componentes que efetivamente implementam os modelos matemáticos do planejamento de *Releases* e das preferências do especialista no domínio com relação a alocação dos requisitos. Além disso, é elaborada uma formulação do PR como problema de otimização interativa, de maneira que o processo de busca possa ser influenciado pela opinião humana.

4.1 VISÃO GERAL

Para possibilitar a resolução do Planejamento de *Releases* como problema de otimização, proporcionando uma efetiva interação entre o processo de geração das soluções e o tomador de decisão (doravante também identificado como usuário), este trabalho propõe uma abordagem composta por três componentes: um Gerenciador de Interações, uma Base de Preferências e um Mecanismo de Otimização. A Figura 6 mostra a organização destes componentes e identifica resumidamente o fluxo do processo de resolução do problema.

Figura 6 – Componentes da abordagem, tipo, sentido e sequência das informações trocadas entre eles e entre o usuário e o Gerenciador de Interações



Fonte: Elaborada pelo autor.

Na referida ilustração, as cores indicam o tipo de informação, o sentido das setas indica qual componente (eventualmente o usuário) envia e recebe as informações e os números representam a ordem de envio das informações. No caso dos números iguais, significa que as respectivas informações são independentes e podem ser enviadas ao mesmo tempo. A seguir será apresentado um resumo da caracterização de cada componente.

a) **Gerenciador de Interações:** este componente tem a responsabilidade de viabilizar todas as interações do usuário. Através dele são definidas algumas configurações iniciais, manipuladas as preferências (inseridas, removidas ou alteradas), visualizadas as melhores soluções e iniciado ou finalizado o processo de busca por soluções;

b) **Base de Preferências:** nesse componente são armazenadas todas as preferências do usuário de tal forma que seja facilitada a obtenção de dados relevantes, como por exemplo, a quantidade de preferências presentes na base e que são satisfeitas por uma determinada solução;

c) **Mecanismo de Otimização:** este componente possui um algoritmo de busca responsável por prover soluções para o problema, ao tempo que as preferências armazenadas na Base de Preferências sejam utilizadas para guiar o processo de busca.

O Algoritmo 2 mostra a dinâmica de execução da abordagem proposta para resolução do Planejamento de *Releases*.

Algoritmo 2: Funcionamento da abordagem

```
Entrada: Instância do planejamento
Saída: A alocação dos requisitos no conjunto de releases disponíveis
início
    Usuário define configurações;
    Usuário informa suas preferências (caso já as possua) ;
    repita
        Executa processo de otimização;
        Usuário visualiza a melhor solução e estado da Base de Preferências;
        se Usuário aceitar a melhor solução então
            Finaliza a busca;
            retorna Melhor solução;
        fim
        senão
            Usuário pode manipular (inserir, alterar, remover) suas preferências;
        fim
    até usuário aceitar uma solução;
fim
```

Considerando como entrada, uma instância contendo as informações definidas na Seção 4.2, relativas ao planejamento das *releases* de um software, o usuário deve utilizar o Gerenciador de Interações para definir alguns parâmetros de configuração e, caso deseje, as suas preferências à cerca da alocação dos requisitos.

As preferências definidas são armazenadas no componente Base de Preferências. Vale destacar a possibilidade do tomador de decisão ainda não possuir nenhuma preferência nesse momento ou ter, mas não desejar expressá-las antes de visualizar uma solução candidata. Por esse motivo, a abordagem é apta para gerar soluções mesmo sem preferências do usuário. Após as interações iniciais, o usuário inicia efetivamente o processo de busca, o qual, será encerrado quando ele aceitar uma solução como sendo a que mais o satisfaz.

O processo de busca por soluções é iniciado com a execução do Mecanismo de Otimização, onde existe um algoritmo de busca especialmente desenvolvido para este fim. São

então geradas soluções considerando-se as variáveis da instância e as informações oriundas da Base de Preferências. O critério de parada e outras características de execução do algoritmo de busca serão detalhadas no Capítulo 5.

Logo após o final da execução do mecanismo de busca, o usuário visualiza através do Gerenciador de Interações, tanto a melhor solução encontrada quanto o estado atual da Base de Preferências. Nesse momento, ele pode aceitar a solução apresentada e encerrar o processo de busca ou executar o algoritmo de busca novamente. No segundo caso, ele pode decidir manipular suas preferências (adicionar, remover ou alterar o nível de importância) na expectativa de que a próxima melhor solução seja mais adequada aos seus anseios.

4.2 MODELAGEM DO PLANEJAMENTO DE *RELEASE*

Conforme relatado no referencial teórico, diversas modelagens para o Planejamento de *Releases* já foram propostas, o que as distinguem, de modo geral, são os fatores considerados por cada uma delas. A seguir, serão explicitados os conjuntos, elementos e atributos utilizados para modelar os fatores estudados neste trabalho, bem como a representação de solução adotada.

4.2.1 Requisitos e *Releases*

O conjunto $R = \{r_1, r_2, r_3, ..., r_N\}$ é tido como o conjunto dos N requisitos que devem ser alocados em P *releases*, representadas por $K = \{k_1, k_2, k_3, ..., k_P\}$. Considerando que todo requisito r_i possui um custo e um risco de implementação a ele associados, $custo_i$ e $risco_i$ são utilizados, respectivamente, para definir tais características. O custo de implementação é um valor (geralmente dado em horas de trabalho ou montante financeiro) que representa o consumo total dos recursos necessários para implementação do requisito. Assim como em (BRASIL, 2011), o risco é definido em termos da análise de impacto do risco para o negócio do cliente *versus* a sua probabilidade de ocorrência. O Quadro 1 mostra a quantificação do risco utilizada neste trabalho.

Os requisitos podem ainda possuir determinados relacionamentos entre si, os quais restringem a aceitação de soluções que não satisfaçam tais condições. Nesta abordagem são consideradas as relações de *precedência* e de *acoplamento* entre as funcionalidades. No primeiro caso, a implementação de um requisito r_i está condicionada a implementação de outro requisito r_j. Já uma relação de acoplamento indica que dois requisitos distintos devem ser implementados

Quadro 1 – Quantificação do Risco através da Análise de Impacto *versus* Probabilidade de Ocorrência

Impacto no Negócio	**Probabilidade de Ocorrência**		
	Baixo	Médio	Alto
Baixo	1	2	3
Médio	4	5	6
Alto	7	8	9

Fonte: Elaborado pelo autor

numa mesma *release*. Assim, ambos os relacionamento podem ser representados pela matiz binária $M_{N\times N}$ que representa todos os N requisitos com seus respectivos dependentes, da seguinte forma:

$$M = \begin{pmatrix} m_{1,1} & m_{1,2} & \cdots & v_{1,i} \\ m_{2,1} & m_{2,2} & \cdots & v_{2,i} \\ \vdots & \vdots & \ddots & \vdots \\ m_{j,1} & m_{j,2} & \cdots & v_{N,N} \end{pmatrix},$$

onde, se $m_{ij} = 1$, o requisito r_i depende do requisito r_j e quando $m_{ij} = m_{ji} = 1$, os requisitos r_i e r_j devem obrigatoriamente ser implementados numa mesma *release*.

Além disso, cada *release* k_q possui uma restrição de orçamento, aqui denotada por s_q. Em face tais definições, as soluções para o Planejamento de *Releases* devem favorecer a alocação antecipada dos requisitos com risco elevado sem, no entanto, desrespeitar as limitações do orçamento de cada *release*.

4.2.2 Clientes

Posto que muitos projetos de software possuem vários interessados no produto final, consideremos $C = \{c_1, c_2, c_3, ..., c_M\}$ como sendo o conjunto de tais envolvidos, onde M é o total destes. Dessa forma, a matriz $V_{m\times n}$ é definida para armazenar o valor de importância que cada interessado c_j estabelece para cada requisito r_i, as colunas representam os M interessados e os N requisitos são representados pelas linhas, como segue:

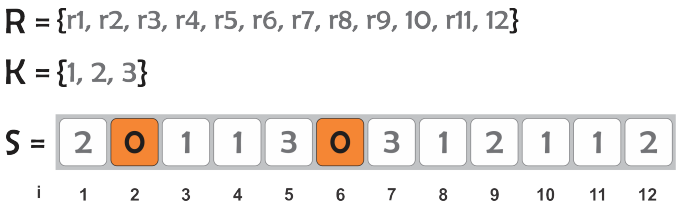
$$V = \begin{pmatrix} v_{1,1} & v_{1,2} & \cdots & v_{1,n} \\ v_{2,1} & v_{2,2} & \cdots & v_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ v_{m,1} & v_{m,2} & \cdots & v_{M,N} \end{pmatrix}.$$

Além disso, é utilizado um fator de peso w_j, associado a cada interessado c_j, para mensurar o nível de importância que ele possui junto à organização responsável pelo desenvolvimento do software.

4.2.3 Representação da Solução

Tendo as definições dos conjunto dos requisitos e das *releases* acima expostas, a abordagem da presente pesquisa adota a representação de uma solução para o planejamento das *releases* como sendo um vetor $S = \{x_1, x_2, x_3, \cdots, x_N\}$, onde $x_i \in \{0, 1, 2, \cdots, P\}$. Se $x_i = 0$, significa que o requisito r_i não está alocado em *release* alguma, caso contrário, o requisito está alocado na *release* k_q, sendo $q = x_i$. A Figura 7 exemplifica um cenário no qual os requisitos r_2 e r_6 não estão alocadas, enquanto os requisitos r_1, r_9 e r_{12} estão selecionado para implementação na release K_2.

Figura 7 – Representação de solução para Planejamento de *Releases* com vetor de inteiros



Fonte: Elaborado pelo autor.

Conforme expõe-se na própria fundamentação teórica deste trabalho, esta representação de solução, bem como os outros aspectos da modelagem aqui apresentados já foram utilizados em outras abordagens de SBSE para o PR e, portanto, não constituem contribuições dessa pesquisa.

4.3 MODELAGEM DA OPINIÃO DO TOMADOR DE DECISÃO

A modelagem matemática da opinião humana constitui-se em um importante desafio para as técnicas de SBSE aplicadas à problemas que requerem participação de um tomador de decisão durante o processo de busca. Segundo Aljawawdeh, Simons e Odeh (2015b) as preferências do usuário podem ser tanto avaliações completamente subjetiva e implícitas, quanto informações explicitamente fornecidas pelo ser humano. No caso do modelo de interação adotado na presente pesquisa, a opinião do usuário da abordagem será expressada de forma explícita.

4.3.1 Modelo para definição de tipos de preferências para o Planejamento de *Releases*

Sabendo que o Planejamento de *Releases* é uma atividade que compreende muitas variáveis, as possibilidade de interação também são diversas. Por isso, a presente pesquisa estabelece um modelo para que tipos de preferências possam ser formalizados e, assim, facilitar tanto o processo de articulação da opinião por parte do usuário, como a captura das informações para o devido uso no algoritmo de busca.

O modelo para definição de tipo de preferências consiste em definir formalmente cada tipo de preferência através de assertivas contendo: uma representação, um conjunto de parâmetros, uma interpretação básica e a representação formal. Este último elemento permite verificar se uma dada solução para o Planejamento de *Release* satisfaz uma determinada preferência do usuário tomador de decisão, relativa ao respectivo tipo de preferência. O Quadro 2 apresenta a descrição de cada um dos elementos necessário à definição de tipos de preferências.

Quadro 2 – Elementos para modelagem de tipos de preferências para Planejamento de *Releases*

Representação	Uma indicação textual e única do tipo da preferência. Contém o nome da preferência e os parâmetros por esta requeridos.
Parâmetros	São elementos proveniente da modelagem do problema, sobre os quais será definida uma opinião, e/ou valores que definam a preferência.
Interpretação Básica	Define o sentido semântico da preferência, ou seja, é a descrição da opinião.
Representação Formal	Define matematicamente os critérios para satisfação do respectivo tipo de preferência. Este elemento é completamente dependente da representação de solução adotada para o problema.

Fonte: Elaborado pelo autor

Além daquilo que já foi dito, a concepção desse padrão para definição de preferências, é também motivado pela flexibilidade e extensibilidade que o mesmo oferece. Com tal modelo é possível criar conjuntos de preferências relacionados a diferentes aspectos do PR, por exemplo, um conjunto de preferências relacionadas aos custos das *releases*, outro com preferências relativas aos *stakeholders* e assim por diante.

4.3.2 Tipos de preferências modelados

A partir do modelo para definição de preferências, foram levantados e modelados 8 tipos de preferências, todos relacionadas à alocação de requisitos em *releases*. Assim, considerando a representação de solução apresentada na Seção anterior, as preferências *Acoplamento Junto*, *Acoplamento Separado*, *Posicionamento Precedente*, *Posicionamento Sucedente*, *Posicionamento Antes*, *Posicionamento Depois*, *Posicionamento Absoluto* e *Posicionamento Excludente*, são formalmente definidas da seguinte forma:

a) *Acoplamento Junto*
   - **Representação:** $coupling\_joint(r_i, r_j)$;
   - **Parâmetros:** Requisitos $r_i \in R$ e $r_j \in R$;
   - **Interpretação Básica:** Dois requisitos distintos devem ser alocados numa mesma *release*;
   - **Representação Formal:** $coupling\_joint(r_i, r_j)$ é satisfeito se, e somente se, $x_i = x_j$.

b) *Acoplamento Separado*
   - **Representação:** $coupling\_disjoint(r_i, r_j)$;
   - **Parâmetros:** Requisitos $r_i \in R$ e $r_j \in R$;
   - **Interpretação Básica:** Dois requisitos distintos devem ser alocados em *releases* diferentes;
   - **Representação Formal:** $coupling\_disjoint(r_i, r_j)$ é satisfeito se, e somente, se $x_i \neq x_j$.

c) *Posicionamento Precedente*
   - **Representação:** $positioning\_precedes(r_i, r_j, [distance])$;
   - **Parâmetros:** Requisitos $r_i \in R$, $r_j \in R$ e uma distância mínima $distance$ entre os requisitos, com valor sempre maior que zero;
   - **Interpretação Básica:** Um requisito r_i deve ser alocado antes do requisito r_j

distante uma certa quantidade de *releases*;

- **Representação Formal:** $positioning\_precedes(r_i, r_j, [distance])$ é satisfeito se pelo menos uma das condições abaixo for atingida:
  - $x_i, x_j \neq 0$ e $x_j - x_i \geq distance$;
  - $x_i \neq 0$ e $x_j = 0$.

d) *Posicionamento Sucedente*

- **Representação:** $positioning\_follows(r_i, r_j, [distance])$;
- **Parâmetros:** Requisitos $r_i \in R$, $r_j \in R$ e uma distância mínima *distance* entre os requisitos, com valor sempre maior que zero;
- **Interpretação Básica:** Um requisito $r_i \in R$ deve ser alocado depois do requisito $r_j \in R$ distante uma certa quantidade de *releases*;
- **Representação Formal:** $positioning\_follows(r_i, r_j, [distance])$ é satisfeito se pelo menos uma das condições abaixo for atingida:
  - $x_i, x_j \neq 0$ e $x_i - x_j \geq distance$;
  - $x_i = 0$ e $x_j \neq 0$.

e) *Posicionamento Depois*

- **Representação:** $positioning\_after(r_i, k_q)$;
- **Parâmetros:** Requisito $r_i \in R$ e uma *release* $q_k \in \{1, 2, 3, \cdots P\}$;
- **Interpretação Básica:** Um requisito r_i deve ser alocado depois de uma certa *release*;
- **Representação Formal:** $positioning\_after(r_i, k_q)$ é satisfeito se pelo menos uma das condições abaixo for atingida:
  - $x_i \neq 0$ e $x_i - k_q \geq 1$;
  - $x_i = 0$.

f) *Posicionamento Antes*

- **Representação:** $positioning\_before(r_i, k_q)$;
- **Parâmetros:** Requisito $r_i \in R$ e uma *release* $q_k \in \{1, 2, 3, \cdots P\}$;
- **Interpretação Básica:** Um requisito r_i deve ser alocado antes de uma certa *release*;
- **Representação Formal:** $positioning\_before(r_i, k_q)$ é satisfeito se, e somente se, $x_i \neq 0$ e $k_q - x_i \geq 1$;

g) *Posicionamento Absoluto*

- **Representação:** $positioning\_in(r_i, k_q)$;
- **Parâmetros:** Requisito $r_i \in R$ e uma *release* $q_k \in \{1, 2, 3, \cdots P\}$;
- **Interpretação Básica:** Um requisito r_i deve ser alocado numa determinada *release*;
- **Representação Formal:** $positioning\_in(r_i, k_q)$ é satisfeito se, e somente se, $x_i = k_q$.

h) *Posicionamento Excludente*

- **Representação:** $positioning\_no(r_i, k_q)$;
- **Parâmetros:** Requisito $r_i \in R$ e uma *release* $q_k \in \{0, 1, 2, 3, \cdots P\}$;
- **Interpretação Básica:** Um requisito r_i deve ser alocado numa *release* diferente de q_k. Caso seja informado 0 como *release*, significa que o requisito r_i deve ser implementado em alguma *release*;
- **Representação Formal:** $positioning\_no(r_i, k_q)$ é satisfeito se, e somente se, $x_i \neq q_k$.

Como é perceptível, os tipos de preferências acima apresentados têm inspiração nos relacionamentos naturais que podem ocorrer entre os requisitos ou entre requisitos e *releases*. O Quadro 3 sintetiza as relações produzidas por cada tipo de preferência.

Quadro 3 – Resumo das relações produzidas pelos tipos de preferências modelados

	RELEASE	**REQUISITO**	
REQUISITO	Antes, Depois, Absoluto, Excludente	Precedente, Sucedente	**Posicionamento**
		Junto, Separado	**Acoplamento**

Fonte: Elaborado pelo autor

Como pode ser visto, se o usuário decidir expressar uma preferência do tipo *f) Posicionamento Antes* ele estará definindo uma relacionamento entre um requisito r_i e uma *release* k_q. Todavia, destaque-se que esta modelagem não tem por função estabelecer restrições capazes de invalidar soluções que não as satisfaçam, mas sim, atuar como restrições leves, aptas a conduzir o mecanismo de busca por regiões do espaço de soluções que sejam mais preferíveis ao tomador de decisão.

Não obstante às informações já detalhadas, durante o processo de planejamento das *releases* de um software, fatores subjetivos influenciam as decisões do profissional responsável por tal atividade. Dessa forma, a tendência natural é que suas preferências possuam diferentes

graus de urgência, ou seja, a satisfação de uma determinada preferência pode ser mais desejada do que a satisfação de uma outra. Por esse motivo, no presente estudo é definido para cada preferência, além do tipo e dos dados que o compõem, um certo nível de importância que representa o quão relevante é o atendimento de tal preferência para o referido especialista. Esse nível de importância varia de 1 (Pouco Importante) à 10 (Altamente Importante), possibilitando assim, uma distinção de relevância entre as preferências.

Diante do que fora exposto à cerca das preferências do usuário, seja considerado $T = \{t_1, t_2, t_3, \cdots, t_Z\}$ como o conjunto de todas as preferências expressas pelo tomador de decisão, com relação à alocação dos requisitos nas *releases*, onde Z é o total de preferências. Sendo que, cada elemento $t_i =< PreferenceAssertion, L >$ é uma tupla constituída por dois elementos, ambos explicitamente definidos pelo usuário da abordagem proposta. O primeiro, *PreferenceAssertion*, é a opinião propriamente dita, que contém a representação do tipo da preferência mais os valores dos parâmetros necessários. O segundo elemento, L, modela o nível de importância da respectiva preferência.

4.4 FORMULAÇÃO INTERATIVA PARA PLANEJAMENTO DE *RELEASES*

Conforme mencionado no referencial teórico, uma condição para tratar um problema como problema de otimização é a existência de uma função capaz de avaliar soluções candidatas distinguindo-as qualitativamente. Assim, e em consonância com os objetivos apresentados no capítulo de Introdução, é proposta uma função de avaliação de soluções que considera tanto os fatores próprios do Planejamento de *Releases*, isto é, as informações da instância do problema, quantos aqueles adicionados pelas interações do usuário, ou seja, as preferências do mesmo.

Assumindo a representação de solução para o Planejamento de *Releases* apresentada na seção 4.2 e as definições relacionadas ao tratamento das preferências do usuário, descritas na seção 4.3, a função de avaliação para uma solução candidata S é definida como sendo:

$$Fitness(S) = \begin{cases} score(S), & se\ Z = 0 \\ \dfrac{score(S)}{penalidade(S,T)} & \text{caso contrário.} \end{cases} \tag{4.1}$$

A função $Fitness(S)$ é flexível para funcionar havendo ou não preferências do usuário a serem consideradas. Isso é justificado pelo fato de ser possível que num dado momento do processo de busca por soluções, a base de preferências esteja vazia, seja porque o usuário preferiu

iniciar a busca antes de definir suas preferências ou porque ele tenha excluído-as no decorrer das interações.

Quando não existem preferência a serem consideradas, ou seja, se $Z = 0$, o valor da função $Fitness(S)$ é dado apenas pela avaliação dos fatores relativos à modelagem do Planejamento de *Releases*, detalhados na seção 4.2 e contemplados pela função $score(S)$ da seguinte forma:

$$score(S) = \sum_{i=1}^{N} (valor_i \times (P - x_i + 1) - risco_i \times x_i) \times y_i, \tag{4.2}$$

onde $y_i \in \{0,1\}$ é uma variável de decisão que contém 1 se o requisito r_i estiver alocado em algum *release*, ou seja, $x_i \neq 0$, e 0 caso contrário. A função $valor_i$ retorna a soma ponderada das importâncias que cada cliente c_j especificou para o requisito r_i, calculada da seguinte forma:

$$valor_i = \sum_{j=1}^{M} w_j \times V(c_j, r_i). \tag{4.3}$$

Percebe-se que a função $score(S)$ é um somatório das composições dos fatores próprios do Planejamento de *Releases* adotados neste trabalho, de forma que o valor dessa função é maior à medida que requisitos mais importantes e com menores riscos são alocados nas primeiras *releases*. Assim, consegue-se maximizar o valor de negócio da alocação e minimizar o risco global do projeto. O princípio dessa fórmula é o mesmo apresentado por Souza *et al.* (2011).

No momento em que existirem preferências a serem consideradas, ou seja, quando o usuário expressar alguma predileção à cerca da alocação dos requisitos, a função $Fitness(S)$ é penalizada de acordo com o nível de importância de cada preferência não satisfeita por uma dada solução S, assim, o processo de evolução tende a privilegiar as soluções mais adaptadas em atender as expectativas do usuário. Tal penalização é dada pela função $penalidade(S)$ da seguinte forma:

$$penalidade(S,T) = 1 + \mu \times \left(\frac{\sum_{i=1}^{Z} L_i \times violation(S,t_i)}{\sum_{i=1}^{Z} L_i} \right). \tag{4.4}$$

Nesta equação, o valor constante 1 é utilizado para evitar uma divisão por 0 (zero) na Equação 4.1, situação que aconteceria em casos nos quais nenhuma das preferências na base fossem satisfeitas pela solução em avaliação. O parâmetro $\mu \in \mathbb{R}^+$, o qual é informado

pelo usuário antes do início do processo de otimização, pondera o quão impactantes são as preferências do usuário na avaliação de aptidão das soluções, ou seja, esta variável estabelece o quanto a Base de Preferências é considerada durante o processo de geração das soluções. A variável L_i é o nível de importância da preferência t_i e $violation(S,t_i)$ verifica se a solução S transgride (não satisfaz) a preferência t_i (Função 4.5). Essa verificação é realizada observando-se os critérios de satisfação estabelecidos na Representação Formal do tipo da preferência t_i.

$$violation(S,t_i) = \begin{cases} 0, & se\ S\ satisfaz\ t_i \\ 1 & \text{caso contrário.} \end{cases} \tag{4.5}$$

Notadamente, quanto maior a quantidade de preferências mais importantes não satisfeitas, maior é o valor da penalidade aplicada. A ideia de preferências mais importantes é obtida a partir do nível de importância L_i de cada preferência T_i. Essa estratégia estimula a priorização da satisfação das preferências que são mais relevantes para o usuário.

Finalmente, a formulação interativa para o Planejamento de *Releases* proposta nesta pesquisa consistem em:

$$\begin{aligned} &\text{maximizar} \quad Fitness(S), \\ &\text{sujeito a:} \quad 1)\ \sum_{i=1}^{N} custo_i \times f_{i,q} \leqslant s_q, \forall q \in \{1,\cdots,P\}, \\ &\qquad\qquad 2)\ x_i \geq x_j, \forall i,j \mid m_{ij} = 1, \end{aligned}$$

onde $f_{i,q}$ indica se o requisito r_i está alocado na *release* k_q e m_{ij} indica se r_i depende de r_j. Assim, 1) representa a restrição de orçamento, assegurando que a soma dos custos dos requisitos alocados para cada *release* k não ultrapasse o orçamento s_k disponível e 2) garante as restrições de precedência e acoplamento.

Diante dos detalhes apresentados nesta seção, a expectativa é que tal formulação seja capaz de incentivar a geração de soluções que contemplem satisfatoriamente as preferências do tomador de decisão, mas também mantenha a otimização dos aspectos próprios do Planejamento de *Releases*.

4.5 EXEMPLO DE APLICAÇÃO DA ABORDAGEM PROPOSTA

Para exemplificar o uso da abordagem proposta, consideremos um cenário fictício no qual uma determinada empresa de desenvolvimento de software precisa desenvolver uma

aplicação web para 3 clientes que atuam no comércio de *FastFood* e prestam serviço de entrega à domicílio.

Após o levantamento de requisitos e negociações com os clientes, ficou definido que o software seria liberado em três versões, tendo a organização desenvolvedora um orçamento de R$ 225,00 para cada período correspondente às *releases*. Naturalmente, os clientes possuem opiniões divergentes em relação à urgência de implementação das funcionalidades e, por consequência, atribuíram diferenciados valores de importância para cada requisito. Ademais, por diferentes razões, tais *stakeholders* possuem relevâncias distintas para a empresa de desenvolvimento. A Tabela 1 apresenta mais detalhes sobre esse contexto.

Tabela 1 – Exemplo de informações que compõem uma instância do Planejamento de *Releases*.

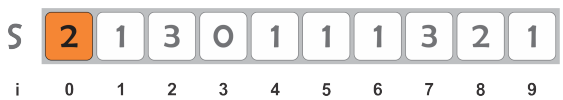
3 *Releases*. Orçamento: R$ 225.00 cada		Importância		3	4	2	
#	**Descrição**	**Custo**	**Risco**	**Cliente 1**	**Cliente 2**	**Cliente 3**	**Total**
0	Efetuar Pedido	R$ 60,00	3	10	10	5	25
1	Buscar Produtos	R$ 40,00	6	8	10	6	24
2	Sugestão de Estabelecimentos Próximos	R$ 40,00	2	6	4	8	18
3	Definir várias Formas de Pagamento	R$ 30,00	6	5	9	1	15
4	Compartilhar nas Redes Sociais	R$ 20,00	4	7	7	5	19
5	Emitir *Feedback* sobre o uso do aplicativo	R$ 20,00	8	8	6	2	16
6	Login através de Redes Sociais	R$ 25,00	9	6	6	4	16
7	Executar em Múltiplas Plataformas	R$ 70,00	7	9	8	3	20
8	Sistema de Recomendação	R$ 50,00	6	6	7	5	18
9	Cadastro de Produtos	R$ 20,00	6	10	10	7	27

Fonte: Elaborado pelo autor.

Diante do cenário apresentado, o gerente de projetos da empresa precisa tomar a decisão sobre quais requisitos devem ser alocados em cada uma das três *releases* da aplicação. Todavia, considerar todas as informações levantadas e realizar uma análise manual seria uma tarefa exaustiva e fatigante. Assim, o referido profissional poderia optar por utilizar, como ferramenta para auxiliá-lo neste processo, a abordagem proposta neste trabalho.

Ao fazer uso da técnica descrita nas seções anteriores deste capítulo, o gerente de projetos se deparou com a solução representada pela Figura 8.

Figura 8 – Representação de possível solução para o Planejamento de *Releases*.



Fonte: Elaborado pelo autor.

Apesar da solução *S* ter sido gerada levando-se em conta todas as informações da Tabela 1 e o algoritmo gerador ter buscado reduzir o risco do projeto e maximizar a satisfação dos clientes respeitando o orçamento limitado, o tomador de decisão percebeu que o requisito 0 (Efetuar Pedido) foi alocado na *release* 2, mas ele preferiria que o fosse na primeira versão. Assim, o profissional decide informar uma preferência do tipo *Posicionamento Absoluto* com a representação *positioning_in(0,1)*, indicando a preferência do mesmo pela alocação do requisito 0 na *release* 1. Após executar novamente o processo de busca, o usuário obtém a solução representada pela Figura 9.

Figura 9 – Representação de possível solução para o Planejamento de *Releases*, considerando preferências do tomador de decisão.

Fonte: Elaborado pelo autor.

Assim como a solução *S*, a nova solução *Si* considera a otimização dos aspectos próprios do planejamento de releases, porém, além disso, esta solução interativa considerou a opinião do profissional para apresentá-lo uma alternativa mais desejável. Evidentemente, o tomador de decisão poderia perceber alguma especificidades da solução corrente e conceber novas preferências, de modo que o processo pararia quando ele julgasse satisfeito.

4.6 CONCLUSÕES DO CAPÍTULO

Nesse Capítulo foram apresentados os fundamentos de concepção da abordagem proposta. Inicialmente, foram apontados a Base de Preferências, o Mecanismo de Otimização e o Gerenciador de Interações, como os componentes responsáveis pela efetiva implementação da técnica proposta.

Através da modelagem matemática de alguns dos fatores importantes para o Planejamento de *Releases*, foi estabelecido quais aspectos seriam objetivamente considerados pela abordagem proposta.

A fim de modelar a opinião do tomador de decisão, foram levantados 8 tipos de preferências, as quais, envolviam as relações entre requisitos e entre requisito e *release*. A partir disso, cada tipo de preferência foi formalmente definido como sendo composto por uma representação textual, um conjunto de parâmetros, uma interpretação básica e a interpretação

formal através da qual expressam-se os critérios de satisfação. Além do tipo, cada preferência expressada pelo usuário possui também um valor de importância, o qual é usado para fazer distinção de relevância entre as preferências.

De posse das modelagens dos aspectos próprios do problema e daqueles advindos da opinião humana, foi elaborada uma formulação do Planejamento de *Releases* como problema de otimização interativa considerando ambas as modelagens para avaliação de soluções candidatas.

Por fim, para exemplificar a aplicabilidade de abordagem proposta, apresentou-se um cenário fictício e como a técnica poderia ser utilizada para realizar o planejamento nas circunstâncias apontadas.

O Capítulo seguinte mostrará todo o projeto de experimento da abordagem proposta, incluindo discussões sobre os resultados alcançados e as constatações obtidas.

5 ESTUDO EMPÍRICO

Este capítulo apresenta os detalhes referentes ao estudo empírico conduzido para avaliação da abordagem proposta. Após a descrição das questões de pesquisa destacadas na primeira seção, na Seção 5.2 é apresentado como foram projetados os experimentos, com informações sobre as instâncias utilizadas, geração de preferências, algoritmo de busca aplicado e as distinções entre os experimentos automático e com humanos. A Seção 5.3 esclarece as métricas elaboradas para interpretação dos resultados, assim como apresenta os dados obtidos e as respectivas discussões relativas a ambos os experimentos.

5.1 QUESTÕES DE PESQUISA

Como forma de avaliar a abordagem proposta, foi conduzido um estudo experimental delineado pelas seguintes questões de pesquisa:

QP1: Qual a capacidade da abordagem em satisfazer as preferências do usuário?

QP2: Qual a consequência da satisfação das preferências do usuário?

QP3: A abordagem proposta é capaz de priorizar o atendimento das preferências mais importantes?

QP4: Qual a relação entre a quantidade de preferências na Base de Preferências e o percentual de preferências atendidas na solução final?

QP5: Diante de uma solução gerada interativamente, quão satisfeito o tomador de decisão estaria em relação ao atendimento de suas preferências?

QP6: Considerando uma solução interativa, qual a relação entre a satisfação do tomador de decisão e o percentual de importância das preferências atendidas?

Para elucidar tais questionamentos, foram realizados teste automatizados e com seres humanos a partir de instâncias artificias e reais do problema de Planejamento de *Releases*. Além disso, para o experimento automático foi gerado, de forma aleatória, um conjunto de preferências para cada instância. Em ambos os testes o algoritmo aplicado como mecanismo de busca foi um Algoritmo Genético. Estes e outros aspectos do experimento serão detalhados nas próximas seções.

5.2 DEFINIÇÕES DOS EXPERIMENTOS

5.2.1 Instâncias

As instâncias “Processador de Texto” e “ReleasePlanner”, utilizadas nos testes, foram constituídas a partir de dados reais dos projetos de desenvolvimento de dois softwares distintos, um processador de textos e uma ferramenta de suporte à decisão para planejamento de *release*. As instâncias originais foram usadas e disponibilizadas por Karim e Ruhe (2014).

O risco de implementação de cada requisito, que originalmente não existia, foi manualmente definido por um desenvolvedor e acrescentado às referidas instâncias. O número de *releases* adotado foi 5 e 8 para “Processador de Texto” e “ReleasePlanner”, respectivamente, valores estes obtidos de forma aleatória considerando o intervalo [5,10]. A justificativa para esta adequação foi aumentar as possibilidades de variação das preferências do usuário, pois o valor original de 3 *releases* para ambas as instâncias limitaria a definição de preferências não conflitantes ou não redundantes.

Visando uma maior diversificação dos dados utilizados nos experimentos, além das instâncias anteriores, mais duas outras nomeadas “dataset-3” e “dataset-4” foram utilizadas, estas porém, tiveram todos os seus dados definidos de forma aleatória, seguindo uma probabilidade uniforme para geração de valores dentro de intervalo pré-definidos. A quantidade de clientes foi gerada aleatoriamente entre 1 e 10, assim como os pesos de cada cliente e as importâncias dos requisitos. O valor de risco foi gerado entre 1 e 9. Com relação ao custo de implementação, o intervalo de valores foi [1,100].

Para as quatro instâncias, o orçamento máximo de cada *release* foi definido como sendo o somatório dos custos de todos os requisitos a serem alocados, dividido pelo número de *releases* da respectiva instância, da seguinte forma:

$$s_q = \left\lfloor \frac{\sum_{i=1}^{N} custo_i}{P} \right\rfloor$$

onde s_q representa o orçamento da *release* q, $custo_i$ o custo de implementação de cada requisito i, N é o total de requisitos e P o total de *releases*.

O Quadro 4 apresenta as quantidades de requisitos, clientes, *releases* e preferências relacionadas à cada instância. Todas estas informações estão formatadas e disponíveis online [1]

[1] Página web com material de suporte desta pesquisa <http://goes.uece.br/altinodantas/masterthesis>

no material de suporte deste trabalho.

Quadro 4 – Informações relacionadas às instâncias utilizadas nos experimentos.

	Requisitos	**Clientes**	***Releases***	**Preferências**
Processador de Texto	50	4	5	50
ReleasePlanner	25	9	8	25
dataset-3	100	3	6	100
dataset-4	150	6	8	150

Fonte: Elaborado pelo autor.

É fundamental destacar que a coluna **Preferências** apresenta a quantidade máxima de preferências geradas para cada instâncias e que estas foram utilizadas apenas no experimento automático. Tal experimento será detalhado mais adiante e o procedimento de geração das referidas preferências será apresentado a seguir.

5.2.2 Geração automática de preferências

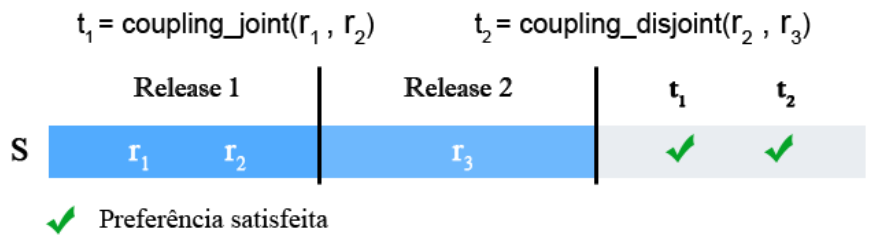
Para simular as preferências do usuário durante o experimento automático, um conjunto de preferências foi gerado de forma aleatória para cada instância utilizada. A quantidade de elementos, ou seja, o total de preferências de cada conjunto foi igual a quantidade de requisitos da respectiva instância. Para uma análise mais precisa do comportamento da abordagem, foram geradas apenas "preferências não conflitantes".

Dada a definição formal das preferências, apresentada na Seção 4.3, duas ou mais preferências são ditas não conflitantes se for logicamente possível que ambas sejam satisfeitas por uma mesma solução. Por exemplo, considerando 2 *releases*, os requisitos r_1, r_2, r_3 e as preferências $t_1 = coupling\_joint(r_1, r_2)$ (a qual requer os dois primeiros requisitos juntos numa mesma *release*) e $t_2 = coupling\_disjoint(r_2, r_3)$ (que requer r_2 e r_3 implementados em *releases* distintas), é possível que ambas as preferências sejam satisfeitas por uma solução S e portanto não há conflito entre as preferências, como mostrado na Figura 10.

Contudo, para atender uma preferência $t_3 = coupling\_joint(r_1, r_3)$ requerendo que r_1 e r_3 fossem implementados numa mesma *release*, seria necessário que t_1 ou t_2 não fossem satisfeitas, caracterizando assim um conflito. Esta situação é apresentada na Figura 11.

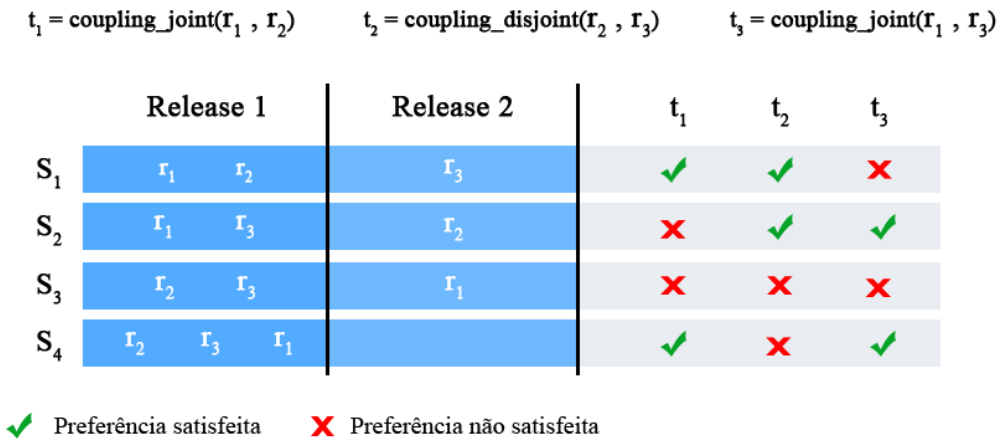
Isto posto, o Algoritmo 3 apresenta a estratégia adotada para a geração automática de preferências não conflitantes, que basicamente, consistiu em criar um conjunto vazio de preferências e iterativamente preenchê-lo até que seu tamanho fosse igual ao total de requisitos

Figura 10 – Exemplo de solução que satisfaz preferências não conflitantes.



Fonte: Elaborado pelo autor.

Figura 11 – Exemplo de soluções para um conjunto de preferências conflitantes.



Fonte: Elaborado pelo autor.

da instância correspondente. Em cada iteração, uma nova preferência foi definida através da geração aleatória dos valores que a compõem, e em seguida, o mecanismo de otimização foi executado com uma configuração de $\mu = 1000$ (essa configuração indica um altíssimo peso da Base de Preferências na avaliação da solução), considerando as preferências antigas e a recém criada. Assim, a nova preferência era aceita como não conflitante sempre que todas tivessem sido satisfeitas por pelo menos uma solução gerada durante a busca.

5.2.3 Algoritmo de busca

A técnica de busca empregada no processo de otimização foi um Algoritmo Genético Interativo adaptado da versão mono-objetivo canônica presente no *framework jMetal*, proposto por Durillo e Nebro (2011). Os parâmetros do algoritmo foram definidos de forma empírica a partir de testes preliminares. O procedimento de cruzamento foi o *"one point"* com probabilidade de 90%. A mutação aplicada com probabilidade de 1%, foi similar ao *"bit flip"*, porém, em consequência da representação de solução adotada, ao invés de inverter o bit, troca-se o valor

Algoritmo 3: Geração de preferências não conflitantes

Entrada: Instância A
Saída: Conjunto T de preferências não conflitantes
início
  Cria o conjunto de preferências T vazio;
  enquanto $|T| <$ *Qtd. de Requisitos de A* **faça**
    Cria preferência t_{atual} vazia;
    Define aleatoriamente o tipo de t_{atual};
    Considerando A, define aleatoriamente os parâmetros da preferência de acordo com o tipo de t_{atual};
    Define aleatoriamente o nível de importância de t_{atual};
    Executa processo de otimização para A considerando T e t_{atual};
    se $\forall t \in T$ *e* t_{atual} *forem satisfeitos e* $t_{atual} \notin T$ **então**
      Adiciona t_{atual} em T;
    fim
  fim
  retorna T
fim

corrente por um valor inteiro aleatoriamente definido dentro do domínio das variáveis de decisão. Como operador de seleção parental foi utilizado torneio binário e a taxa de elitismo foi de 1%.

Além dos operadores canônicos do AG, um operador de reparo foi utilizado para garantir a satisfação da restrição de custo associada a cada *release*. Essa reparação consiste em: para cada *release* com custo de implementação maior do que o orçamento disponível, até que a *release* respeite esta restrição, seleciona-se aleatoriamente um requisito nela presente e verifica-se a possibilidade de inclusão de tal requisito em outra *release* de modo que não extrapole o orçamento desta *release* de destino. Se for possível, faz-se a transferência do requisito e o consequente ajuste dos custos de ambas as *releases*. Caso contrário, o requisito é simplesmente marcado como não alocado e seu custo de implementação deduzido do custo total de implementação da *release* em reparo.

Com relação a restrição de precedência técnica entre requisitos, foi aplicada uma severa penalização aos indivíduos inválidos, sendo tal penalização proporcional a quantidade de precedências não atendidas. A população foi constituída por 100 indivíduos e como critério de parada foi adotado um limite de 1000 gerações ou 200 gerações sem melhoria, o que ocorrer

primeiro.

Em razão da possibilidade do tomador de decisão não ficar totalmente satisfeito com a melhor solução encontrada e decidir reexecutar o processo de busca, foi utilizado um certo mecanismo de ligação entre as execuções consecutivas do AGI. Na prática, esse mecanismo consiste em capturar o indivíduo mais adaptado de uma dada execução e inseri-lo na população inicial da execução seguinte possibilitando que características evoluídas anteriormente sejam aproveitadas no novo processo evolucionário.

5.2.4 Experimento automático

Neste experimento não houve participação humana. O aspecto interativo consistiu na simulação das interações de um tomador de decisão. Para isto, foi utilizado o conjunto de preferências definido para cada uma das instâncias, conforme detalhado na Subseção 5.2.2.

A abordagem foi testada da seguinte forma: para cada uma das quatro instâncias, a técnica foi executada simulando duas interações. Na primeira, um percentual das preferências disponíveis para a respectiva instância era inserido na Base de Preferências e o mecanismo de busca retornava uma solução considerando o estado corrente da base. Em seguida, havia a segunda interação, na qual cada preferência presente na base tinha uma probabilidade de 50% de chance de ser substituída por outra, ou, caso todas as preferências já estivessem na base, o valor de importância da mesma era aleatoriamente alterado seguindo uma probabilidade uniforme dentro do intervalo possível de valores desta propriedade. Evidentemente, após a segunda interação, o algoritmo de busca retornava uma nova solução e esta era então considerada a solução final daquela execução da abordagem.

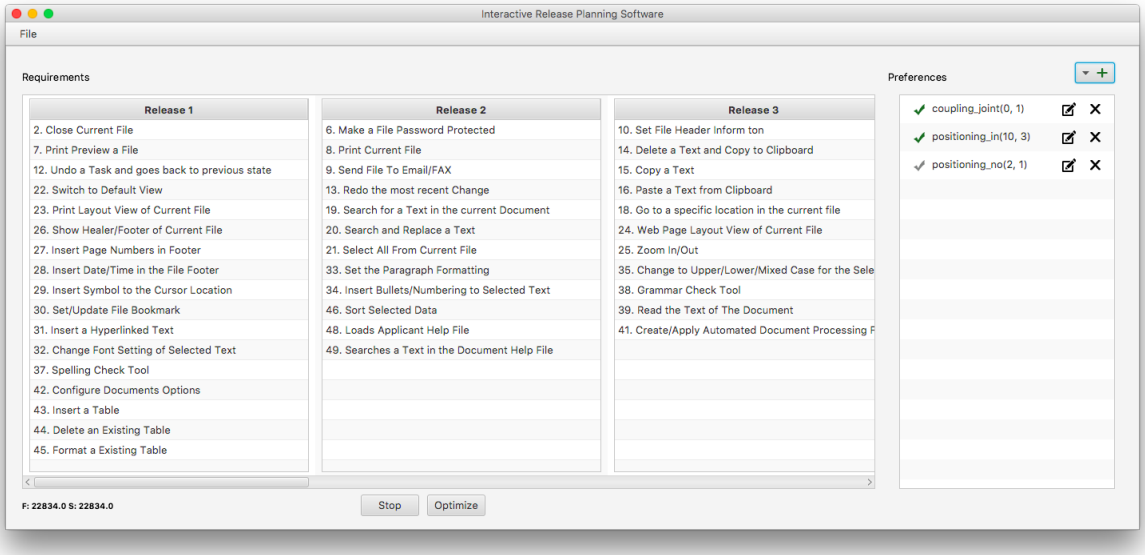
Como a principal motivação para o experimento automático é a possibilidade de realização de testes exaustivos, a abordagem foi executada para as quatro instâncias com todas as combinações das configurações do peso μ, variando de 0 a 1 com intervalos de 0,1 e do percentual de preferências na base que variou de 10% a 100% do total de cada conjunto de preferências, com intervalos de 10%.

Em função do caráter estocástico da meta-heurística utilizada no processo de busca, cada teste foi executado 30 vezes. Dessa forma, foram criadas as amostras submetidas aos testes estatísticos *Wilcoxon signed-rank test* e $\hat{A}_{12}$, ambos realizados através da ferramenta estatística R (R Core Team, 2015).

5.2.5 Experimento com humanos

Para viabilizar o experimento com tomadores de decisão reais, foi desenvolvida uma interface gráfica responsável por implementar as definições do componente Gerenciador de Interações. A Figura 12 apresenta a tela principal da ferramenta produzida.

Figura 12 – Tela principal da ferramenta desenvolvida para utilização da abordagem proposta.



Fonte: Elaborado pelo autor.

O funcionamento da ferramenta consiste em: após ser carregada uma instância do Planejamento de *Releases* através do menu *File>Open* da interface, a primeira solução é apresentada em forma de colunas na parte centro-esquerda da janela. Cada coluna representa uma *release* e dentro desta encontram-se os requisitos para ela selecionados. No lado direito da janela apresentam-se informações relativas à Base de Preferências. Através do botão "+" é possível adicionar uma nova preferência, para isso, deve-se escolher entre os 8 tipos disponíveis e preencher os parâmetros requeridos. Logo abaixo, é apresentada uma lista com todas as preferências presentes na base, inclusive para cada preferência é indicado se esta é ou não satisfeita pela solução corrente. É possível ainda, alterar o nível de importância de uma preferência ou excluí-la da lista. Na parte inferior da interface existem os botões "Stop" e "Optimize". O primeiro tem a função de encerrar o processo de planejamento, já o segundo é responsável por acionar o mecanismo de otimização e, por consequência, ocorre a geração de uma nova solução na área destinada a isto. É importante destacar que para este experimento, o Mecanismo de Otimização foi configurado com um nível de influência da Base de Preferência igual a 1 ($\mu = 1$).

De posse da interface gráfica capaz de assegurar a interação entre o usuário tomador de decisão e o processo de otimização, 10 profissionais com formação acadêmica e experiência em desenvolvimento de software foram convidados para participar deste experimento. O Quadro 5 apresenta informações detalhadas sobre o perfil de cada um dos participantes.

Quadro 5 – Resumo do perfil dos participantes recrutados para o experimento

Participantes	**Q1**	**Q2**	**Q3**	**Q4**	**Q5**
#1	Pós-Graduado	Professor	7 anos	Alta	Média
#2	Pós-Graduado	Professor	8 anos	Média	Média
#3	Pós-Graduado	Professor	17 anos	Média	Baixa
#4	Pós-Graduado	Analista de Sistemas	10 anos	Alta	Média
#5	Pós-Graduando	Analista de Sistemas	3 anos	Alta	Média
#6	Pós-Graduado	Desenvolvedor	4 anos	Media	Média
#7	Pós-Graduando	Desenvolvedor	12 anos	Alta	Média
#8	Graduado	Arquiteto de Software	8 anos	Alta	Média
#9	Pós-graduando	Desenvolvedor/Professor	6 anos	Média	Média
#10	Pós-Graduando	Analista de Testes	6 anos	Média	Média

Fonte: Elaborado pelo autor.

Nota: Q1 (Formação acadêmica), Q2 (Ocupação atual), Q3 (Experiência em TI), Q4 (Experiência em Desenvolvimento de software) e Q5 (Experiência em seleção de requisitos).

Percebe-se que 3 anos é o menor tempo de experiência de um participante, enquanto o participante com mais tempo possui 17 anos; Os 10 participantes têm em média 8,1 anos de experiência com Tecnologia da Informação. Além disso, com exceção do participante #8, todos possuem ou estão cursando pós-graduação relacionada a engenharia de software e metade deles desfruta de “alta” experiência com desenvolvimento.

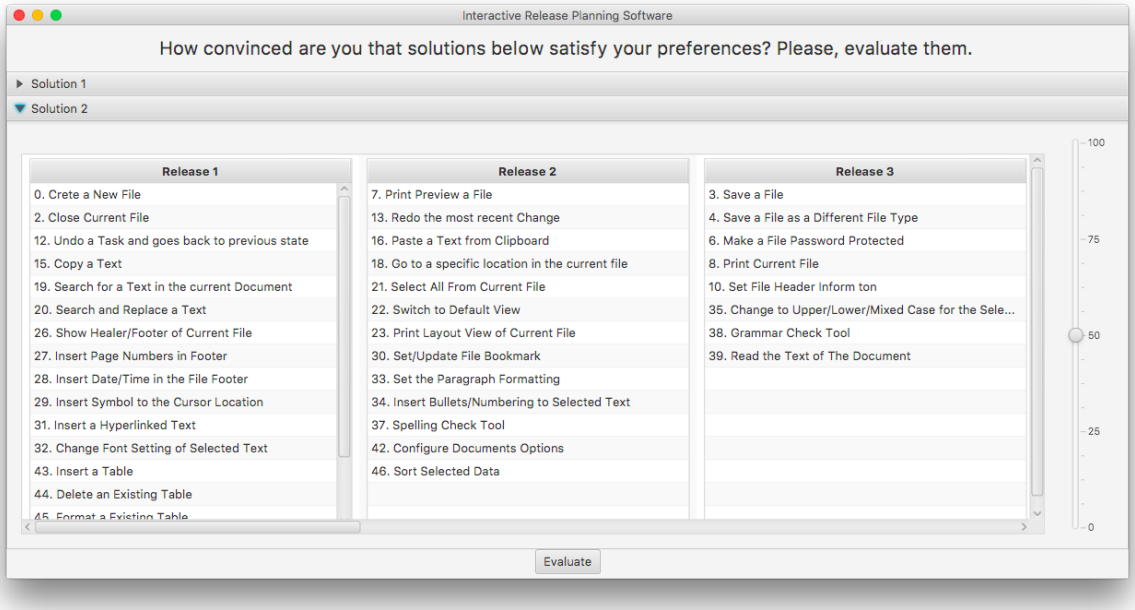
Como procedimento inicial do experimento, cada um dos convidados foi separadamente conduzido a um computador previamente equipado com o material necessário para realização experimento, ou seja, a ferramenta, duas instâncias, um documento explicando cada tipo de preferência modelado e os formulários contendo questionários. Antes de iniciar o teste propriamente dito, o participante recebeu explicações sobre a área de pesquisa na qual este estudo está inserido, o problema em questão, a abordagem proposta e como seria conduzido o restante do processo experimental.

Após estas explicações introdutórias, o participante foi convidado a ler e assinar um termo de consentimento (Apêndice A). Em seguida foi aplicado um questionário de coleta de informações (Apêndice B). Tendo consentido participar do estudo, o convidado recebeu explicações sobre a modelagem das preferências estabelecida na abordagem proposta e, desse modo, foi esclarecido de como poderia expressar suas predileções.

Ciente dos tipos de opiniões suportadas pela abordagem e de como articulá-las, o participante utilizou a ferramenta para resolver uma instância fictícia (a mesma apresentada na Tabela 1). Tal procedimento teve como propósito familiarizar o usuário com a interface e com a dinâmica do experimento. Após o referido teste preliminar, realizou-se o experimento definitivo com a instância real de um Processador de Textos.

Durante o processo de uso da ferramenta, o usuário teve acesso a um documento contendo todas as informações relativas à instância. Ao decidir aceitar uma solução como definitiva para o planejamento da instância em questão, isto é, quando clicasse em "Stop", duas soluções identificadas apenas como Solução 1 e Solução 2 eram apresentadas ao participante, conforme pode ser visto na Figura 13.

Figura 13 – Tela usada para avaliação de soluções durante experimento com profissionais



Fonte: Elaborado pelo autor.

À direita de cada uma das soluções havia um *slider* através do qual o participante era convidado a tentar mensurar, percentualmente, o quão convencido ele estava de que a respectiva solução atendia as expectativas dele. Como nesse momento o usuário já não tinha acesso às informações da Base de Preferências, com dados objetivos, essa avaliação tinha relação com o sentimento de aceitação e, por isso, é considerada uma Avaliação Subjetiva. Na prática, a primeira solução era obtida sem a utilização da Base de Preferências durante o processo de busca e a segunda era a solução interativa que o próprio participante aceitou como ideal.

A seguir serão apresentados os dados resultantes de cada um dos experimentos e também serão conduzidas as respectivas discussões.

5.3 RESULTADOS E ANÁLISES

5.3.1 Métricas

Nesta subseção, serão detalhadas as métricas criadas para facilitar a análise dos resultados e por conseguinte favorecer a resposta das questões de pesquisa.

5.3.1.1 Preferências Satisfeitas (PS)

A métrica *PS* é o percentual unitário de Preferências Satisfeitas por uma determinada solução em comparação ao total de preferências definidas pelo usuário. Assim, dado um conjunto $T = \{t_1, t_2, t_3, \cdots, t_Z\}$ de preferências para uma determinada instância D, esta métrica mensura o quanto uma solução candidata S satisfaz T. Considerando que existe um conjunto $T' \subseteq T$, onde cada elemento de T', ou seja, cada preferência t'_i, é atendida pela solução S, o valor de *PS* é dado por:

$$PS(S) = \frac{|T'|}{|T|}.$$

Considere um cenário com 3 *releases*, 6 requisitos e um conjunto T com quatro preferências: $t_1 = coupling\_joint(1;6)$ que requer os requisitos r_1 e r_6 juntos numa mesma release, $t_2 = positioning\_in(1;3)$ que requer o requisito r_1 alocado na *release* k_3, $t_3 = coupling\_disjoint(2;3)$ que requer os requisitos r_2 e r_3 alocados em *releases* diferentes e $t_4 = positioning\_follow(4;5;1)$ requerendo que o requisito r_4 seja alocado pelo menos uma *release* depois da *release* na qual r_5 for alocado. Considere também $S = [1,3,1,2,2,3]$ como uma solução para planejamento nas citadas condições. Neste contexto, o conjunto T' seria composto por t_2 e t_3 uma vez que apenas estas preferências são satisfeitas por S, dessa forma $PS(S)$ seria $\frac{2}{4} = 0.5$, ou seja, 50% das preferências do usuário teriam sido satisfeitas.

5.3.1.2 Nível de Satisfação (NS)

A métrica *Nível de Satisfação* é o percentual unitário dos níveis de importância satisfeitos em comparação ao total dos níveis de importância de todas as preferências presentes na base de preferências. Assim, dado um conjunto $T = \{t_1, t_2, t_3, \cdots, t_Z\}$ de preferências para

uma determinada instância D e sabendo que o nível de importância de cada preferência representa a urgência que o usuário tem pelo atendimento da respectiva preferência, esta métrica é utilizada para mensurar o quão satisfeitas, por uma solução S, foram as prioridades do usuário. Considerando que existe um conjunto $T' \subseteq T$, onde cada elemento de T', ou seja, cada preferência t'_i, é satisfeita pela solução S, o valor de NS é dado por:

$$NS(S) = \frac{\sum_{t' \in T'} L'_t}{\sum_{t \in T} L_t},$$

onde L é o nível de importância de cada preferência t e t'.

Aproveitando o cenário exemplificado na Subseção 5.3.1.1, seja considerado que as preferências t_1, t_2, t_3 e t_4 possuem 3, 5, 4 e 2 respectivamente como níveis de importância L_t. Dessa forma, se verificada a métrica NS para aquele exemplo no qual apenas as preferências t_2 e t_3 foram satisfeitas, ter-se-ia $\frac{5+4}{3+5+4+2} = 0,642$, ou seja, apesar de terem sido atendidas 50% das preferências, o nível de satisfação chegaria a 64,2%.

5.3.1.3 Ganho em Preferências (GP)

A métrica Ganho em Preferências é o percentual unitário de variação da quantidade de preferências satisfeitas *(PS)* quando, para uma mesma instância, compara-se uma solução gerada com participação humana e outra solução gerada sem considerar as preferências do usuário, da seguinte forma:

$$GP(S, Si) = \left(\frac{PS(Si)}{PS(S)} \right) - 1,$$

onde Si é uma solução interativa, ou seja, gerada considerando-se as preferências do usuário e S uma solução sem influências da opinião humana durante o processo de busca.

5.3.1.4 Preço das Preferências (PP)

A métrica Preço das Preferências é o percentual unitário de variação no valor de *score* quando, para uma mesma instância, compara-se uma solução gerada com participação humana e outra solução gerada sem considerar as preferências do usuário. Apresentada por Araújo (2014) esta métrica mensura o quanto as preferências podem afetar a otimização dos fatores próprios da modelagem do Planejamento de *Releases*, no presente trabalho, o prejuízo é refletido no valor de negócio e risco do projeto. O cálculo de PP é realizado da seguinte forma:

$$PP(S, Si) = 1 - \left(\frac{score(Si)}{score(S)} \right),$$

onde *Si* é uma solução gerada levando-se em consideração as preferências do usuário e *S* uma solução gerada considerando-se apenas a função *score*.

5.3.1.5 Densidade de Preferências (DP)

A métrica Densidade de Preferências é o percentual de preferências para uma determinada instância, com relação à quantidade de requisitos da mesma. Assim, dada uma instância D contendo um conjunto R de requisitos e um conjunto T de preferências relativas a alocação destes em *releases*, DP é mensurada por

$$DP(D) = \frac{|T|}{|R|} \times 100.$$

Para exemplificar a ideia dessa métrica, considere a instância "dataset-3" a qual possui 100 requisitos. Se num dado momento durante o processo de resolução do planejamento, a base possuir 20 preferências sobre a alocação dos requisitos, nesse momento a densidade de preferências seria $\frac{20}{100} \times 100$, ou seja, 20%.

5.3.2 Experimento automático

A seguir serão apresentados os resultados e analisados os dados relativos ao experimento automático compreendendo as questões de pesquisa **QP1**, **QP2**, **QP3** e **QP4**. Por limitação de espaço nem todas os dados deste experimento estão aqui dispostos, contudo, estes podem ser consultados online na página de suporte[2] da pesquisa .

5.3.2.1 Análise da influência das preferências nas soluções finais

A Tabela 2 mostra a média dos valores de *PS* obtidos em 30 execuções, destaca os resultados do teste $\hat{A}_{12}$ comparando os valores de *PS* de cada configuração de μ com os valores obtidos sem considerar as preferências, ou seja, quando $\mu = 0$. Além disso, apresenta a indicação da diferença estatística entre os resultados consecutivos de cada configuração de μ. Os dados relatados são pertinentes às quatro instâncias e a três diferentes níveis de densidade de preferências (*DP*), alto = 100%, médio = 50% e baixo 10%.

A primeira linha de cada um dos valores de DP, aquela na qual $\mu = 0$, corresponde aos valores de *PS* obtidos sem considerar da Base de Preferências ao longo do processo de busca,

[2] <http://goes.uece.br/altinodantas/masterthesis>

Tabela 2 – Média dos resultados de *PS*, valores do teste $\hat{A}_{12}$ entre cada configuração de μ com relação à configuração $\mu = 0$, e, indicação de diferença estatística ao se variar μ, para cada instância utilizando-se diferentes quantidade de preferências

		Processador de Textos			**ReleasePlanner**			**dataset-3**			**dataset-4**		
DP	μ	PS	$\hat{A}_{\mu 0}$		PS	$\hat{A}_{\mu 0}$		PS	$\hat{A}_{\mu 0}$		PS	$\hat{A}_{\mu 0}$	
	0	0,371	-	-	0,379	-	-	0,465	-	-	0,444	-	-
	0,1	0,442	**0,86**	▲	0,508	**0,97**	▲	0,513	**0,82**	▲	0,471	**0,72**	▲
	0,2	0,501	**0,98**	▲	0,567	**1**	▲	0,560	**0,97**	▲	0,498	**0,91**	▲
	0,3	0,535	**0,99**	▲	0,607	**1**	▲	0,589	**0,99**	▲	0,539	**0,97**	▲
	0,4	0,579	**1**	▲	0,648	**1**	▲	0,635	**1**	▲	0,559	**1**	▲
100%	**0,5**	0,625	**1**	▲	0,675	**1**	▲	0,664	**1**	▲	0,582	**1**	▲
	0,6	0,669	**1**	▲	0,695	**1**	△	0,685	**1**	▲	0,587	**1**	△
	0,7	0,693	**1**	△	0,715	**1**	▲	0,699	**1**	△	0,608	**1**	△
	0,8	0,720	**1**	△	0,737	**1**	▲	0,720	**1**	▲	0,613	**1**	△
	0,9	0,725	**1**	△	0,752	**1**	△	0,728	**1**	△	0,622	**1**	△
	1,0	0,739	**1**	△	0,769	**1**	△	0,746	**1**	△	0,639	**1**	▲
	0	0,443	-	-	0,359	-	-	0,519	-	-	0,474	-	-
	0,1	0,575	**0,94**	▲	0,515	**0,96**	▲	0,595	**0,85**	▲	0,548	**0,85**	▲
	0,2	0,709	**1**	▲	0,587	**1**	▲	0,657	**0,97**	▲	0,600	**0,96**	▲
	0,3	0,767	**1**	▲	0,618	**1**	△	0,728	**1**	▲	0,635	**0,99**	▲
	0,4	0,783	**1**	△	0,703	**1**	▲	0,759	**1**	△	0,681	**1**	▲
50%	**0,5**	0,811	**1**	▲	0,738	**1**	▲	0,789	**1**	▲	0,700	**1**	△
	0,6	0,857	**1**	▲	0,754	**1**	△	0,803	**1**	△	0,716	**1**	△
	0,7	0,851	**1**	▽	0,764	**1**	△	0,833	**1**	▲	0,747	**1**	▲
	0,8	0,893	**1**	▲	0,769	**1**	△	0,831	**1**	▽	0,759	**1**	△
	0,9	0,877	**1**	▽	0,769	**1**	*	0,851	**1**	△	0,768	**1**	△
	1,0	0,895	**1**	△	0,772	**1**	△	0,862	**1**	△	0,772	**1**	△
	0	0,513	-	-	0,211	-	-	0,493	-	-	0,529	-	-
	0,1	0,900	**0,96**	▲	0,667	**1**	▲	0,797	**0,98**	▲	0,767	**0,96**	▲
	0,2	1,000	**1**	▲	0,711	**1**	△	0,887	**1**	▲	0,844	**1**	▲
	0,3	1,000	**1**	*	0,867	**1**	▲	0,887	**1**	*	0,873	**1**	△
	0,4	1,000	**1**	*	0,978	**1**	▲	0,920	**1**	▲	0,869	**1**	▽
10%	**0,5**	1,000	**1**	*	1,000	**1**	△	0,903	**1**	▽	0,902	**1**	▲
	0,6	1,000	**1**	*	1,000	**1**	*	0,907	**1**	△	0,904	**1**	△
	0,7	1,000	**1**	*	1,000	**1**	*	0,930	**1**	△	0,904	**1**	*
	0,8	1,000	**1**	*	1,000	**1**	*	0,930	**1**	*	0,942	**1**	▲
	0,9	1,000	**1**	*	0,989	**1**	▽	0,917	**1**	▽	0,924	**1**	▽
	1,0	1,000	**1**	*	1,000	**1**	△	0,930	**1**	△	0,918	**1**	▽

Fonte: Elaborado pelo autor.

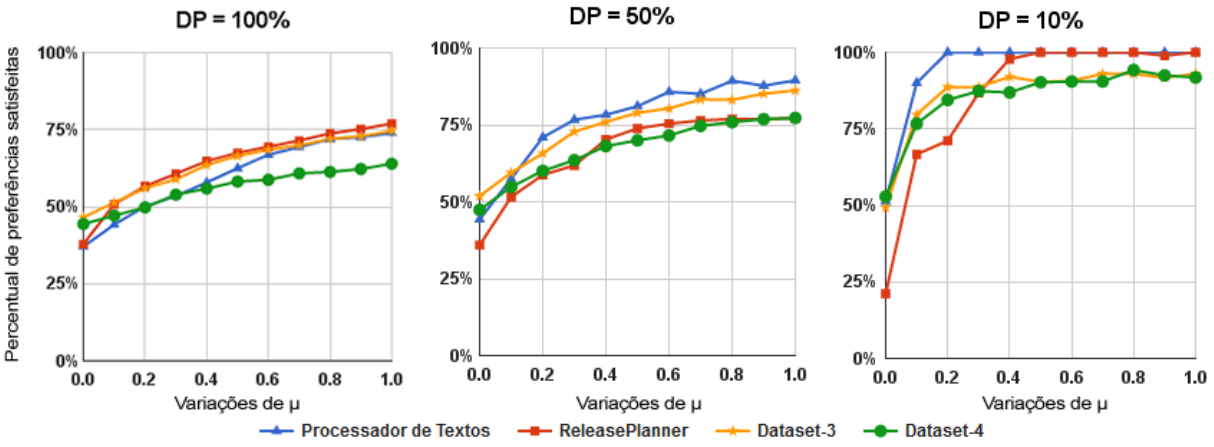
Nota: O símbolo △ indica que o resultado de *PS* é insignificantemente maior do que o *PS* obtido com a configuração de μ imediatamente anterior, ▽ (insignificantemente menor), ▲ (significantemente maior) e ▼ (significantemente menor), tudo isso considerando-se 0,05 como nível de significância. * Indica que respectiva comparação estatística não pôde ser realizada em função das amostras serem idênticas.

ou seja, foram produzidos por soluções geradas sem intervenção do tomador de decisão. Por exemplo, para a instância "Processador de Textos" atingiu-se 0,371, 0,443 e 0,513 de *PS* com 100%, 50% e 10% de densidade de preferências, respectivamente. Tais resultados são explicados

pelo fato de uma dada solução poder, por coincidência, satisfazer algumas preferências. Contudo, observa-se que em nenhum dos casos, de nenhuma das instâncias, esses resultados foram maiores que 0,53 e, em 75% das vezes foram menores que 0,5. Novamente, tomando os dados do "Processador de Textos" mas agora analisando as linhas nas quais $\mu = 0,1$, isto é, resultados obtidos com a menor influência da Base de Preferências (dentre as opções testadas), foram obtidos 0,442, 0,575 e 0,9 para 100%, 50% e 10% de *DP*, respectivamente. Destaque-se o valor de *PS* com 10% de *DP* que passou de 0,371 para 0,9. Ampliando esta análise para todas as instâncias e os três níveis de densidade de preferências, constata-se que em todos os casos, ao se comparar a configuração $\mu = 0,1$ com $\mu = 0$, houve aumento significante da quantidade de preferências satisfeitas, conforme indicação (▲) oriunda do teste de diferença estatística e confirmação de superioridade com magnitude "grande" obtida pelo $\hat{A}_{12}$ ($\hat{A}_{\mu 0} > 0,71$). É interessante destacar que a partir da configuração $\mu = 0,4$, independente da instância e da quantidade de preferências na base, a correspondente estimativa $\hat{A}_{\mu 0}$ foi 1.

Ainda considerando os dados da Tabela 2, a Figura 14 apresenta uma visão mais geral sobre o comportamento de variação da quantidade média de preferências satisfeitas quando se aumenta o nível de influência da base de preferências no processo de busca, ou seja, quando o valor de μ aumenta. Na referida figura, apresentam-se gráficos com as variações dos resultados de cada instância com alta, média e baixa Densidade de Preferências, a média dos valores de *PS* (expressos em porcentagem) obtidos nas quatro instâncias.

Figura 14 – Percentual médio de Preferencias Satisfeitas (PS) de todas as instâncias, considerando a variação de μ e diferentes quantidades de preferências



Fonte: Elaborado pelo autor.

Como pode ser visto, a medida que se aumenta o valor de μ os valores de *PS* obtidos

para todas as instâncias também tendem a crescer. Observa-se que esse crescimento é mais significante nas variações iniciais do peso μ. Isso é naturalmente explicado pelo fato de que, depois de se atingir determinados níveis de satisfação de preferências, um pequeno aumento no peso da Base de Preferências pode não ser suficiente para que o algoritmo consiga satisfazer mais preferências.

Respondendo a **QP1**: "*Qual a capacidade da abordagem em satisfazer as preferências do usuário?*", pode-se afirmar que independente da instância, a abordagem consegue atender mais preferências do usuário do que a versão não interativa avaliada, tanto com baixa, média e alta densidade de preferências, considerandos os tipos de preferências modelados nesta pesquisa. Além disso, para todas as instâncias e níveis de Densidade de Preferências, o percentual de preferências satisfeitas pela abordagem tende a ser maior quanto maior o valor de μ.

5.3.2.2 Análise da consequência de inclusão das preferências do Tomador de Decisão

Para facilitar a análise de como as soluções são impactadas em decorrência do atendimento das preferências do tomador de decisão, foram processados os dados relativos ao *score* das soluções e verificado o comportamento da abordagem quanto a esse aspecto. A Tabela 3 apresenta as médias, tanto de *GP* quanto de *PP*, para todas as instâncias, três níveis de Densidade de Preferências e cada variação de $\mu \neq 0$, considerando 30 execuções de cada combinação. Por definição, não existe Preço da Preferências nem Ganho em Preferências para as combinações que envolvem a configuração $\mu = 0$, uma vez que as soluções geradas com esta configuração não são interativas. Também são apresentadas as indicações de diferença estatística entre os valores de *PP* gerados por variações consecutivas em μ e a estimativa produzida pelo teste $\hat{A}_{12}$ comparando-se os valores de *score* das soluções geradas com o respectivo μ e os *scores* atingidos com $\mu = 0$.

Observando especificamente os dados referentes à instância "*ReleasePlanner*", percebe-se que usando $\mu = 0,1$ ocorreu uma perda de *score* 0,5%, 0,5% e 0,18% para 100%, 50% e 10% de Densidade de Preferências, respectivamente. A propósito, com esta configuração de μ os valores de *PP* não ultrapassaram 0,018 em nenhuma das instâncias e níveis *DP*. Em contrapartida, os ganhos em preferências satisfeitas foram de pelo menos 0,06 (dataset-4 com *DP* = 100%). Analisando a configuração $\mu = 0,5$, nota-se que dentre todas as instâncias e configurações de *DP* o maior valor de *PP* foi 0,058, atingido para o Processador de Textos com *DP* = 50%, enquanto 0,83 foi o *GP* observado com esta mesma combinação, ou seja, para

Tabela 3 – Média dos resultados de *GP* e *PP* obtidos para todas as instâncias, diferentes configurações de μ e variadas quantidades de preferências, indicação de diferença estatística entre os valores consecutivos de *PP* e estimativa do $\hat{A}_{12}$ entre cada configuração de μ com relação à configuração $\mu = 0$ com base nos valores de *score*

		Processador de Textos				**ReleasePlanner**				**dataset-3**				**dataset-4**			
DP	μ	**GP**	**PP**		$\hat{A}_{\mu 0}$	**GP**	**PP**		$\hat{A}_{\mu 0}$	**GP**	**PP**		$\hat{A}_{\mu 0}$	**GP**	**PP**		$\hat{A}_{\mu 0}$
100%	**0,1**	0,19	0,010	▲	**0,68**	0,34	0,005	▲	**0,66**	0,10	0,005	△	0,58	0,06	0,010	▲	0,65
	0,2	0,35	0,015	△	**0,78**	0,50	0,018	▲	**0,85**	0,20	0,007	△	0,60	0,12	0,016	△	**0,7**
	0,3	0,44	0,030	▲	**0,86**	0,60	0,022	△	**0,9**	0,27	0,013	△	**0,68**	0,21	0,004	▼	0,54
	0,4	0,56	0,039	△	**0,93**	0,71	0,042	▲	**0,99**	0,37	0,011	▽	**0,68**	0,26	0,010	△	0,63
	0,5	0,69	0,054	▲	**0,98**	0,78	0,051	▲	**0,99**	0,43	0,024	▲	**0,86**	0,31	0,014	△	0,67
	0,6	0,80	0,069	▲	**0,99**	0,83	0,061	▲	**1**	0,47	0,030	△	**0,87**	0,32	0,018	△	**0,7**
	0,7	0,87	0,071	△	**0,99**	0,89	0,072	▲	**1**	0,50	0,039	▲	**0,94**	0,37	0,022	△	**0,78**
	0,8	0,94	0,089	▲	**1**	0,95	0,080	△	**1**	0,55	0,042	△	**0,96**	0,38	0,029	△	**0,85**
	0,9	0,96	0,093	△	**1**	0,99	0,093	△	**1**	0,56	0,046	△	**0,97**	0,40	0,038	△	**0,89**
	1,0	0,99	0,105	△	**1**	1,03	0,099	△	**1**	0,60	0,060	▲	**0,99**	0,44	0,036	▽	**0,89**
50%	**0,1**	0,30	0,009	▲	**0,67**	0,44	0,005	△	0,64	0,15	*-0,006*	▽	0,44	0,16	*-0,007*	▽	0,45
	0,2	0,60	0,026	▲	**0,84**	0,64	0,015	▲	**0,83**	0,27	0,011	▲	0,68	0,27	*-0,008*	▽	0,43
	0,3	0,73	0,041	▲	**0,9**	0,72	0,019	△	**0,81**	0,40	0,012	△	**0,72**	0,34	*-0,003*	△	0,47
	0,4	0,77	0,051	△	**0,96**	0,96	0,038	▲	**0,99**	0,46	0,017	△	**0,83**	0,44	0,003	△	0,58
	0,5	0,83	0,058	△	**0,98**	1,06	0,049	▲	**1**	0,52	0,026	△	**0,85**	0,48	0,008	△	0,65
	0,6	0,94	0,062	△	**0,98**	1,10	0,057	△	**1**	0,55	0,034	△	**0,93**	0,51	0,015	△	**0,71**
	0,7	0,92	0,065	△	**0,98**	1,13	0,060	△	**1**	0,61	0,041	△	**0,95**	0,58	0,017	△	0,70
	0,8	1,02	0,081	▲	**1**	1,14	0,057	▽	**1**	0,60	0,036	▽	**0,94**	0,60	0,022	△	**0,75**
	0,9	0,98	0,077	▽	**0,99**	1,14	0,060	△	**1**	0,64	0,041	△	**0,94**	0,62	0,028	△	**0,79**
	1,0	1,02	0,089	△	**1**	1,15	0,055	▼	**1**	0,66	0,056	▲	**0,97**	0,63	0,028	△	**0,81**
10%	**0,1**	0,75	0,015	▲	**0,71**	2,16	0,018	▲	**0,89**	0,61	0,002	△	0,50	0,45	0,000	▽	0,52
	0,2	0,95	0,024	△	**0,81**	2,37	0,023	△	**0,92**	0,80	0,000	▽	0,50	0,60	*-0,006*	▽	0,45
	0,3	0,95	0,020	▽	**0,78**	3,11	0,028	△	**0,96**	0,80	0,003	△	0,53	0,65	0,004	△	0,56
	0,4	0,95	0,019	▽	**0,79**	3,63	0,034	△	**0,98**	0,86	0,005	△	0,60	0,64	*-0,007*	▽	0,44
	0,5	0,95	0,017	▽	**0,76**	3,74	0,034	△	**1**	0,83	0,011	△	**0,68**	0,71	*-0,001*	△	0,51
	0,6	0,95	0,021	△	**0,79**	3,74	0,038	△	**1**	0,84	0,005	▽	0,60	0,71	0,001	△	0,54
	0,7	0,95	0,024	△	**0,85**	3,74	0,038	▽	**1**	0,89	0,010	△	0,63	0,71	0,002	△	0,53
	0,8	0,95	0,020	▽	**0,82**	3,74	0,036	▽	**1**	0,89	0,005	▽	0,57	0,78	0,010	△	0,63
	0,9	0,95	0,022	△	**0,8**	3,68	0,039	△	**1**	0,86	0,012	△	**0,66**	0,75	0,003	▽	0,55
	1,0	0,95	0,024	△	**0,83**	3,74	0,036	▽	**1**	0,89	0,004	▽	0,53	0,74	0,002	▽	0,52

Fonte: Elaborado pelo autor.

Nota: O símbolo △ indica que o resultado de Preço das Preferências é insignificantemente maior do que o *PP* obtido com a configuração de μ imediatamente anterior, ▽ (insignificantemente menor), ▲ (significantemente maior) e ▼ (significantemente menor), tudo isso considerando-se 0,05 como nível de significância. Na coluna $\hat{A}_{\mu 0}$, os valores em negrito indicam que ocorreu diferença estatística entre os valores de *score* atingidos com o respectivo μ e os obtidos com $\mu = 0$, também considerando $\alpha = 0,05$.

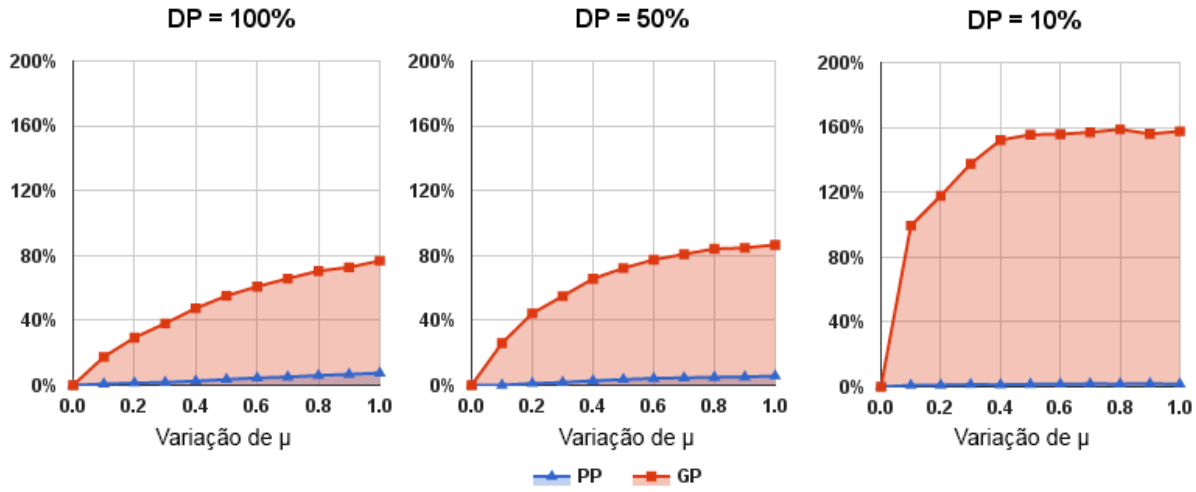
aumentar em 83% a quantidade de preferências atendidas perdeu-se 5,8% em *score*.

É natural ocorrer perda de *score* quando se aumenta a quantidade de preferências atendidas, pois as opiniões do tomador de decisão podem conflitar com outros aspectos matemáticos do planejamento, porém, em 7 situações (valores em itálico) ocorreram resultados negativos

para esta perda, ou seja, na realidade ocorreu um ganho. Recorrendo aos dados da coluna $\hat{A}_{\mu 0}$ (que mostra em negrito a estimativa de superioridade entre amostras com diferença estatística) percebe-se que na prática não há diferenças significantes entre os *scores* das soluções com *PP* negativo e os *scores* das correspondentes soluções não interativas ($\mu = 0$). Ainda relacionado à comparação dos *scores* das soluções interativas ($\mu \neq 0$) com os das soluções não interativas para a mesma instância e com mesmo *DP*, em todos os casos com significante diferença estatística, a estimativa do $\hat{A}_{12}$ foi no mínimo "média" em favor das soluções interativas, indicando assim, a probabilidade de perda nos *score* de tais soluções.

Atentando para os dados de *PP*, excetuando-se os referentes à configuração $\mu = 0$, em 95 ocasiões (quase 80% dos casos) ocorreu aumento em relação ao valor da configuração imediatamente anterior, entretanto, em apenas 28 delas (29%) houve diferença estatística significante entre resultados de *PP* para valores consecutivos de μ. Essa constatação é um indicativo de que, apesar de haver uma certa tendência de progressão da perda de *score* à medida que se aumenta a influência da Base de Preferências no processo de busca por soluções, o valor de *score* não é tão sensível a pequenas variações de μ. A Figura 15 apresenta o comportamento tanto da *PP* quanto do *GP* quando μ varia. Para isso, calculada a média dos dados das quatro instâncias em cada nível de Densidade de Preferências.

Figura 15 – Comparação entre *GP* e *PP*, considerando a média dos resultados obtidos para as quatro instâncias com cada densidade de preferências e configuração do μ



Fonte: Elaborado pelo autor.

Como observa-se, o preço das preferências tende a crescer com o aumento do parâmetro μ, contudo, numa proporção muito menor do que o crescimento verificado no ganho

percentual de preferências atendidas. Por exemplo, com *DP* = 100% na configuração $\mu = 0$ os resultados para *PP* e *GP* são 0, porém, com $\mu = 1$ os valores são 7% e 77%, respectivamente.

Por fim, respondendo a **QP2**: "*Qual a consequência da satisfação das preferências do usuário?*" pode-se afirmar que há uma perda no *score* das soluções para que a opinião do usuário seja considerada e consequentemente suas preferências atendidas. Todavia, observando as quatro instâncias, níveis alto, médio e baixo de Densidade de Preferências e as variações de $\mu \neq 0$, a perda matemática nos critérios próprios do Planejamento de *Releases* não chegou a 11% em comparação com soluções não-interativas.

5.3.2.3 Análise da priorização das preferências mais importantes

Em razão da modelagem das preferências estabelecer que cada preferência possui um certo nível de importância, definido que o tomador de decisão possa distingui-las em termos de prioridade, foi realizada uma avaliação quanto a relação entre o percentual de preferências satisfeitas e o percentual do somatório dos níveis de importância atingido pela abordagem. A Tabela 4 apresenta as médias de *PS* e *NS* de 30 execuções para cada instância, configuração de μ e nível de Densidade de Preferências, além da indicação da estimativa de superioridade em resultados correspondentes a uma mesma configuração.

Observando os dados relativos à instância "dataset-3" vê-se que, com $\mu = 0,1$ os resultados de *PS* e *NS* foram respectivamente 0,51 e 0,54 para *DP* = 100%, 0,59 e 0,60 para *DP* = 50% e 0,80 e 0,91 para *DP* = 10%. Nesses três casos as médias de *NS* foram superiores às de *PS*, todavia, na configuração *DP* = 50% não houve diferença estatística significante entre as amostras e, consequentemente, a magnitude da superioridade pode ser considerada desprezível. Por outro lado, analisando a instância "dataset-4", independente da Densidade de Preferências, *NS* foi significantemente maior do que *PS* em todos as configurações de μ, indicando assim que houve uma priorização em satisfazer as preferências mais importantes da referida instância. Vale destacar que das 30 comparações entre *NS* e *PS* do "dataset-4", em 29 delas (96,6%) a magnitude da superioridade de *NS* foi "média" ($\hat{A}_{NSPS} > 0,64$) ou "grande" ($\hat{A}_{NSPS} > 0,71$). Ampliando esta observação para todas as comparações com diferença estatística significante (99 de 120), em 5 ocasiões (cerca de 5% dos casos) a magnitude foi "pequena" em todas as outras foi "média" ou "grande".

Apesar de ter ocorrido diferença estatística em 81,6% de todas as comparações entre *NS* e *PS*, deve-se ressaltar que em 14 situações, todas verificadas com a configuração *DP* =

Tabela 4 – Média dos resultados de *PS*, *NS* e valores de $\hat{A}_{12}$ entre ambos, para variações de μ em cada instância, considerando-se diferentes quantidades de preferências

		Processador de Textos			**ReleasePlanner**			**dataset-3**			**dataset-4**		
DP	μ	PS	NS	$\hat{A}_{NSPS}$	PS	NS	$\hat{A}_{NSPS}$	PS	NS	$\hat{A}_{NSPS}$	PS	NS	$\hat{A}_{NSPS}$
100%	**0,1**	0,44	0,50	**0,79**	0,51	0,51	0,49	0,51	0,54	**0,66**	0,47	0,48	**0,61**
	0,2	0,50	0,58	**0,88**	0,57	0,58	**0,54**	0,56	0,61	**0,78**	0,50	0,52	**0,69**
	0,3	0,53	0,61	**0,82**	0,61	0,63	**0,68**	0,59	0,64	**0,84**	0,54	0,57	**0,69**
	0,4	0,58	0,65	**0,85**	0,65	0,68	**0,66**	0,64	0,70	**0,88**	0,56	0,59	**0,75**
	0,5	0,62	0,68	**0,76**	0,67	0,71	**0,73**	0,66	0,72	**0,86**	0,58	0,62	**0,77**
	0,6	0,67	0,72	**0,72**	0,69	0,74	**0,77**	0,69	0,75	**0,91**	0,59	0,63	**0,78**
	0,7	0,69	0,73	**0,75**	0,71	0,76	**0,82**	0,70	0,77	**0,9**	0,61	0,65	**0,8**
	0,8	0,72	0,76	**0,77**	0,74	0,78	**0,77**	0,72	0,79	**0,95**	0,61	0,66	**0,81**
	0,9	0,73	0,78	**0,76**	0,75	0,80	**0,73**	0,73	0,80	**0,93**	0,62	0,67	**0,81**
	1	0,74	0,79	**0,76**	0,77	0,81	**0,73**	0,75	0,82	**0,9**	0,64	0,68	**0,82**
50%	**0,1**	0,57	0,60	**0,63**	0,52	0,56	**0,73**	0,59	0,60	0,56	0,55	0,58	**0,67**
	0,2	0,71	0,74	**0,65**	0,59	0,64	**0,77**	0,66	0,70	**0,69**	0,60	0,64	**0,68**
	0,3	0,77	0,80	**0,65**	0,62	0,67	**0,63**	0,73	0,79	**0,77**	0,64	0,69	**0,75**
	0,4	0,78	0,83	**0,77**	0,70	0,75	**0,75**	0,76	0,82	**0,85**	0,68	0,74	**0,78**
	0,5	0,81	0,84	**0,68**	0,74	0,78	**0,76**	0,79	0,85	**0,84**	0,70	0,77	**0,88**
	0,6	0,86	0,86	0,54	0,75	0,80	**0,84**	0,80	0,87	**0,89**	0,72	0,79	**0,82**
	0,7	0,85	0,87	**0,64**	0,76	0,80	**0,94**	0,83	0,88	**0,81**	0,75	0,81	**0,79**
	0,8	0,89	0,91	**0,61**	0,77	0,81	**1**	0,83	0,90	**0,85**	0,76	0,82	**0,83**
	0,9	0,88	0,89	0,58	0,77	0,81	**1**	0,85	0,90	**0,81**	0,77	0,84	**0,95**
	1	0,89	0,90	0,58	0,77	0,81	**0,97**	0,86	0,91	**0,86**	0,77	0,84	**0,93**
10%	**0,1**	0,90	0,93	**0,57**	0,67	0,94	**1**	0,80	0,91	**0,9**	0,77	0,88	**0,86**
	0,2	1,00	1,00	0,50	0,71	0,95	**0,88**	0,89	0,96	**0,88**	0,84	0,95	**0,9**
	0,3	1,00	1,00	0,50	0,87	0,98	**0,58**	0,89	0,97	**0,97**	0,87	0,96	**0,88**
	0,4	1,00	1,00	0,50	0,98	1,00	0,50	0,92	0,98	**0,79**	0,87	0,96	**0,93**
	0,5	1,00	1,00	0,50	1,00	1,00	0,50	0,90	0,97	**0,9**	0,90	0,98	**0,97**
	0,6	1,00	1,00	0,50	1,00	1,00	0,50	0,91	0,97	**0,88**	0,90	0,97	**0,88**
	0,7	1,00	1,00	0,50	1,00	1,00	0,50	0,93	0,98	**0,74**	0,90	0,97	**0,88**
	0,8	1,00	1,00	0,50	1,00	1,00	0,50	0,93	0,98	**0,74**	0,94	0,98	**0,69**
	0,9	1,00	1,00	0,50	0,99	1,00	0,50	0,92	0,98	**0,82**	0,92	0,98	**0,85**
	1	1,00	1,00	0,50	1,00	1,00	0,50	0,93	0,98	**0,74**	0,92	0,97	**0,78**

Fonte: Elaborado pelo autor.

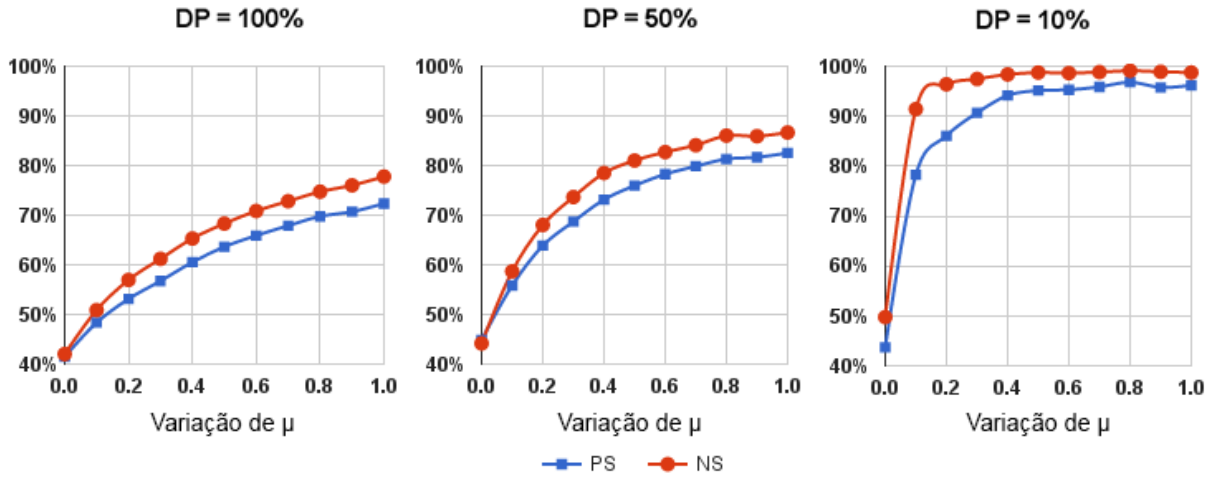
Nota: Na coluna $\hat{A}_{NSPS}$, os valores em negrito indicam que ocorreu diferença estatística entre os valores de *PS* e *NS* da respectiva linha, considerando $\alpha = 0,05$ como nível de significância.

10%, 100% das preferências haviam sido satisfeitas e, portanto, os valores de *NS* e *PS* são necessariamente iguais, justificando assim, não haver diferença estatística entre os resultados das métricas.

A fim de sintetizar o comportamento geral dos resultados de Preferências Satisfeitas

e Nível de Satisfação nas condições avaliadas, a Figura 16 apresenta gráficos para alto, médio e baixo nível de Densidade de Preferências nos quais são apresentadas as médias dos resultados de *PS* e *NS* das quatro instâncias com cada variação de μ.

Figura 16 – Comparação entre *PS* e *NS*, considerando a média dos resultados obtidos para as quatro instâncias com cada densidade de preferências e configuração do μ



Fonte: Elaborado pelo autor.

Como pode ser visto, em média, independente da quantidade de preferências, a tendência é que os valores de *NS* sejam superiores aos de *PS*, evidenciando, assim, que as soluções que satisfazem as preferências mais importantes são priorizadas.

Diante do que foi analisado é possível concluir que a abordagem é capaz de priorizar as preferências mais importantes, uma vez que, considerando todas as instâncias, os níveis de Densidade de Preferências e diferentes níveis de influência da Base de Preferências no processo de busca, em 93,3% dos casos possíveis, as preferências mais importantes foram priorizadas. Considera-se portanto, respondida a **QP3**.

5.3.2.4 Análise do impacto da quantidade de preferências

As análises realizadas nos tópicos anteriores utilizaram dados provenientes de três níveis de Densidade de Preferências, 10% considerado baixo, 50% médio e 100% alto. Contudo, é necessário considerar intervalos menores de variação de *DP* para uma melhor avaliação sobre a relação entre a quantidade de preferências presentes na base e o percentual de atendimento destas pelas soluções geradas com a abordagem proposta.

Nesse sentido, a Tabela 5 apresenta os dados coletados de 30 execuções, conside-

rando as quatro instâncias, três níveis influência da Base de Preferências (μ) e dez diferentes quantidades de preferências. Para subsidiar tal análise há também indicação da diferença estatística em resultados produzidos por configurações subsequentes de *DP*, para instância e μ correspondentes.

Tabela 5 – Média dos resultados de *PS* e indicação de diferença estatística entre os valores obtidos para cada instância e cada variação de densidade de preferências, considerando-se três diferentes configurações de μ

		Densidade de Preferências (DP)									
μ	**Instâncias**	**10%**	**20%**	**30%**	**40%**	**50%**	**60%**	**70%**	**80%**	**90%**	**100%**
0,1	Processador de Textos	0,90	0,89 ▽	0,74 ▼	0,65 ▼	0,57 ▼	0,53 ▼	0,54 △	0,50 ▼	0,46 ▼	0,44 ▽
	ReleasePlanner	0,67	0,80 ▲	0,72 ▼	0,59 ▼	0,52 ▼	0,56 △	0,58 ▲	0,58 △	0,55 ▽	0,51 ▼
	dataset-3	0,80	0,62 ▼	0,58 ▽	0,62 ▲	0,59 ▽	0,59 ▽	0,57 ▽	0,56 ▽	0,53 ▼	0,51 ▽
	dataset-4	0,77	0,63 ▼	0,58 ▼	0,56 ▽	0,55 ▽	0,51 ▼	0,49 ▼	0,48 ▽	0,47 ▽	0,47 ▽
0,5	Processador de Textos	1,00	0,95 ▼	0,95 △	0,88 ▼	0,81 ▼	0,80 ▽	0,78 ▽	0,75 ▽	0,68 ▼	0,62 ▼
	ReleasePlanner	1,00	0,92 ▼	0,88 ▽	0,80 ▼	0,74 ▼	0,74 △	0,73 ▼	0,71 ▽	0,76 ▲	0,67 ▼
	dataset-3	0,90	0,88 ▼	0,85 ▼	0,81 ▼	0,79 ▼	0,76 ▼	0,75 ▽	0,73 ▽	0,69 ▼	0,66 ▼
	dataset-4	0,90	0,86 ▼	0,78 ▼	0,76 ▽	0,70 ▼	0,68 ▼	0,62 ▼	0,62 ▽	0,59 ▼	0,58 ▽
1	Processador de Textos	1,00	0,97 ▼	0,98 △	0,93 ▼	0,89 ▼	0,86 ▼	0,87 △	0,85 ▽	0,77 ▼	0,74 ▽
	ReleasePlanner	1,00	1,00 –	0,94 ▼	0,81 ▼	0,77 ▼	0,80 ▲	0,79 ▼	0,81 ▲	0,80 ▽	0,77 ▼
	dataset-3	0,93	0,91 ▽	0,90 ▽	0,87 ▼	0,86 ▽	0,85 ▽	0,83 ▽	0,81 ▼	0,79 ▽	0,75 ▼
	dataset-4	0,92	0,93 △	0,88 ▼	0,83 ▼	0,77 ▼	0,73 ▼	0,70 ▼	0,67 ▼	0,66 ▽	0,64 ▼

Fonte: Elaborado pelo autor.

Nota: O símbolo △ indica que o resultado de Nível de Satisfação é insignificantemente maior do que o *NS* obtido com a quantidade de preferências (DP) imediatamente anterior, ▽ (insignificantemente menor), ▲ (significantemente maior) e ▼ (significantemente menor), tudo isso considerando-se 0,05 como nível de significância.

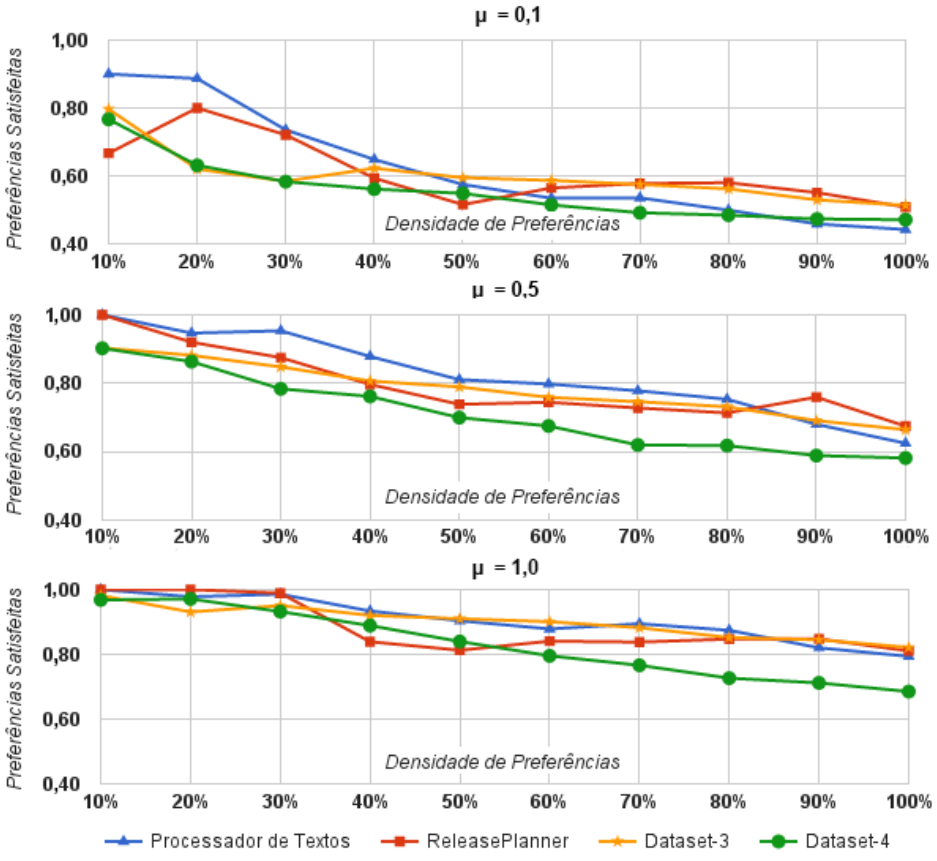
Observando os dados obtido com 10% de Densidade de Preferências, percebe-se que, por exemplo, para a instância "Processador de Textos" 90%, 100% e 100% das preferências foram satisfeitas com $\mu = 0,1$, $\mu = 0,5$ e $\mu = 1$, respectivamente. Com este valor de *DP*, os resultados de *PS* foram todos iguais ou superiores 0,8, com exceção da instância "ReleasePlanner" com $\mu = 0,1$. Passando para a configuração *DP* = 20%, para a instância "Processador de Texto" obteve-se 0,89, 0,95 e 0,97, para $\mu = 0,1$, $\mu = 0,5$ e $\mu = 1$, respectivamente. Isso significa que independente da configuração de μ houve redução no percentual de preferências satisfeitas quando aumentada a quantidade destas. Tal comportamento é similar para todas as instâncias, as únicas exceções são "RealeasePlanner" com $\mu = 0,1$ e "dataset-4" com $\mu = 1$.

Analisando toda a variação de *DP* (10% a 100%) para a instância "Processador de Textos" especificamente na configuração $\mu = 0,1$, nota-se que em 6 de 9 ocasiões possíveis, houve significativa diminuição (▼) na quantidade de preferências satisfeitas, quando comparados

valores atingidos por configurações consecutivas de *DP*. Ampliando a análise para todas as instâncias com $\mu = 0,1$, em 47,2% dos casos foi verificada redução estatisticamente significante. Considerando a configuração *DP* = 0,5 as reduções foram significativas em 63,8% das comparações e para *DP* = 1 ocorreram em 52,7% dos casos. De forma geral, em 54,6% das vezes em que houve um aumento de 10% em Densidade de Preferências, ocorreu também significativa diminuição na quantidade de preferências atendidas.

Para uma visão mais geral de como os resultados se apresentam com cada um dos níveis de influência da Base de Preferência (μ), a Figura 17 mostra o comportamento dos valores de *PS* ao longo das variações na Densidade de Preferências.

Figura 17 – Nível de Satisfação obtido para cada instância considerando variação da quantidade de preferências com três diferentes configurações de μ



Fonte: Elaborado pelo autor.

Vê-se que em cada uma das configurações de μ, os resultados de *PS* para todas as instâncias são mais próximos de 1 para os valores menores de Densidade de Preferências e tendem a diminuir com o aumento em *DP*. Nota-se também que os valores iniciais com a configuração $\mu = 1$ são por mais vezes iguais a 1. Como já demonstrado em análise anterior,

isso é exatamente uma consequência de um peso maior para a influência da Base de Preferências durante o processo de geração das soluções. Não obstante, o comportamento não crescente de *PS* é também verificado nas outras configurações de μ, mesmo sem iniciar com valores próximos a 100% de preferências satisfeitas.

Retomando a **QP4**: "*Qual a relação entre a quantidade de preferências na Base de Preferências e o percentual de preferências atendidas na solução final?*", é possível afirmar quanto mais opiniões o tomador de decisão expressar mais difícil é satisfazê-lo por completo, pois independente do nível de influência da Base de Preferências para o mecanismo de busca, o percentual de preferências atendidas pelas soluções, tende a ser menor à medida que se aumenta a quantidade de preferências a serem consideradas.

5.3.3 Experimento com Humanos

Para avaliação do comportamento da abordagem quando utilizada por profissionais da área de desenvolvimento de software, foram colhidos diversos dados referentes ao teste de cada um dos 10 participantes.

A Tabela 6 apresenta informações relacionadas a 1) interação usuário-abordagem, como a quantidade de vezes que o tomador de decisão executou o processo de otimização, o tempo necessário para realização do planejamento e quantidade final de preferências na base; 2) Resultados das métricas *PS*, *NS* e *PP* tanto para a solução não interativa quanto para a solução final aceita pelo usuário; e 3) Avaliação subjetiva que o próprio tomador de decisão forneceu para a primeira e última solução.

Como pode ser visto, foi bastante variada a quantidade de vezes nas quais, buscando por uma nova solução, cada participante reexecutou o Mecanismo de Otimização. Por exemplo, enquanto o participante #1 realizou 8 tentativas e ao final estava com 3 preferências na base, o #2 chegou a 109 reexecuções e 4 preferências. Em média os participantes precisaram de 35 tentativas até encontrar uma solução suficientemente satisfatória. Ressalte-se que, os dois participantes que mais tentaram novas soluções não possuíam "Alta" experiência com desenvolvimento de software. Com relação ao tempo decorrido desde a visualização da primeira solução até uma solução ser aceita, passando por todo processo de interações, os participantes gastaram em média 23,04 minutos. Nesse sentido, os três participantes que passaram de 25 minutos também possuíam "Média" experiência na área de desenvolvimento. Sabendo que a instância "Processador de Textos" utilizada no teste possui 50 requisitos, a densidade final de preferências foi no máximo

Tabela 6 – Resultados do teste para cada um dos 10 participantes

Participantes	**#1**	**#2**	**#3**	**#4**	**#5**	**#6**	**#7**	**#8**	**#9**	**#10**	**Média**
Qtd. de buscas	8	109	15	47	23	54	20	31	20	25	$35{,}20 \pm 29{,}48$
Tempo em minutos	16,14	45,40	26,93	19,57	11,81	25,41	21,50	19,88	16,87	26,88	$23{,}04 \pm 9{,}26$
Qtd. final de preferências	3	4	8	11	4	10	33	20	20	5	$11{,}80 \pm 9{,}70$
PS (*Si*)	1,00	1,00	0,88	0,91	1,00	0,90	0,78	1,00	0,85	1,00	$0{,}93 \pm 0{,}08$
PS (*S*)	0,33	0,75	0,25	0,55	0,50	0,30	0,27	0,30	0,25	0,60	$0{,}41 \pm 0{,}18$
NS (*Si*)	1,00	1,00	0,90	0,88	1,00	0,90	0,76	1,00	0,86	1,00	$0{,}93 \pm 0{,}08$
NS (*S*)	0,33	0,75	0,27	0,52	0,43	0,31	0,28	0,28	0,23	0,61	$0{,}40 \pm 0{,}17$
PP (*Si*)	0,07	0,02	0,04	0,04	0,01	0,09	0,07	0,13	0,03	0,006	$0{,}05 \pm 0{,}04$
Avaliação Subjetiva (*Si*)	0,933	0,873	0,875	0,750	0,933	0,843	0,746	0,876	0,502	0,870	$0{,}820 \pm 0{,}128$
Avaliação Subjetiva (*S*)	0,652	0,814	0,498	0,248	0,621	0,575	0,624	0,128	0,2478	0,00	$0{,}441 \pm 0{,}266$

Fonte: Elaborado pelo autor.

Nota: (*Si*) é uma indicação de que o respectivo valor refere-se a uma solução interativa, ou seja, foi gerada com influência da Base de Preferências e no caso desse experimento foi utilizado $\mu = 1$. (S) é uma indicação para resultados de uma solução não-interativa, gerada sem considerar as preferências do participante, ou seja, $\mu = 0$.

de 66% (participante #7), mas em média foi de 23%.

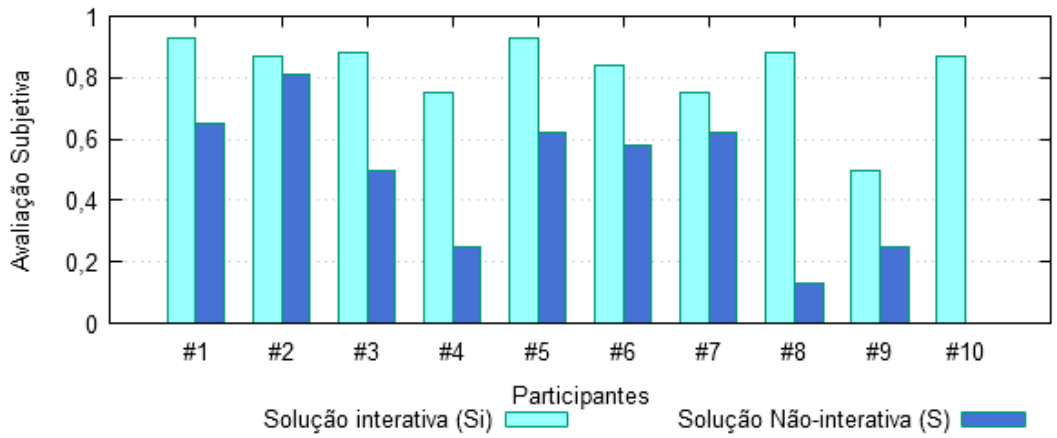
Conforme mencionado antes, a primeira solução *S* exibida pela ferramenta é gerada antes mesmo do usuário começar expressar suas opiniões, ou seja, não é interativa. De posse dessa solução inicial e da solução *Si* construída com ajuda do participante, é possível compará-las através das métricas *PS*, *NS* e *PP*. Nesse sentido, observa-se que, em média, as soluções interativas (*Si*) atenderam 93% das preferências dos participantes, enquanto as não interativas (*S*) atenderiam 41%. Os resultados de *NS* foram em média 93% e 40%, para *Si* e *S*, respectivamente. E, a perda média de *score* das soluções geradas com auxílio dos participantes, em relação à soluções automáticas, foi de 5%. Estes resultados corroboram com as constatações obtidas através do experimento automático, indicando que a abordagem comportou-se de forma similar nos dois experimentos e que as conclusões obtidas no primeiro são válidas também para este.

5.3.3.1 Análise da satisfação dos profissionais

Apesar da importante constatação de que a abordagem possui, na prática, o mesmo comportamento verificado nas simulações, é fundamental que o usuário da ferramenta sinta-se satisfeito quanto as soluções geradas interativamente. Nesse sentido, os resultados mostram que os participantes #1 e #5 avaliaram a solução gerada com a participação dos mesmos como mais que 93% satisfatória, sendo que 70% dos participantes atribuíram uma avaliação superior a 0,8. Destaca-se que os três participantes mais satisfeitos possuíam "Alta" experiência. O participante #9 foi o único com avaliação abaixo de 0,7, ainda assim, para ele a solução Si (gerada com

ajuda dele) foi mais que duas vezes superior à solução *S* (não interativa) para a qual, o mesmo concedeu uma avaliação subjetiva de 0,24. A Figura 18 apresenta uma visão das avaliações subjetivas entre as soluções interativas e não-interativas avaliadas por cada um dos participantes.

Figura 18 – Avaliações subjetivas para soluções interativas e não-interativas por participante



Fonte: Elaborado pelo autor.

Como pode ser visto, em alguns casos, como para os participantes #2 e #7, a avaliação subjetiva da solução não-interativa chegou a ser próxima à nota atribuída para a solução interativa. Contudo, para todos os participantes, as soluções interativas foram mais satisfatórias do que as soluções não-interativas inicialmente apresentadas. Destaque para o participante #10 que ficou completamente insatisfeito com a solução não-interativa (*S*) e chegou a mais de 80% de satisfação para a solução gerada com influência da opinião do mesmo.

Em face a tudo que fora apresentado, pode-se afirmar em resposta a **QP5**: "*Diante de uma solução gerada interativamente, quão satisfeito o tomador de decisão estaria em relação ao atendimento de suas preferências?*", que, em média, a técnica proposta é capaz de tornar o tomador de decisão 82% satisfeito. Ademais, através do questionário de *feedback* aplicado ao final do experimento, 80% dos participantes responderam "Eficaz" ou "Muito eficaz" quando questionados sobre a eficácia da estratégia de auxiliar o algoritmo durante o processo de geração das soluções. Além disso, 30% afirmaram que com certeza usariam a ferramenta num ambiente real de desenvolvimento, 40% disseram que provavelmente usariam, 20%, possivelmente e o participante #9 declarou que dificilmente usaria a técnica proposta.

5.3.3.2 Análise da relação entre a avaliação subjetiva e o nível de importância das preferências satisfeitas

Conforme mencionado anteriormente, por intermédio do *NS* é possível mensurar objetivamente o quanto uma determinada solução *S* atinge do total de importância de todas as preferências presentes na base, o que é realizado pela verificação do nível de importância de cada preferência satisfeita por *S*. Com essa característica, o *NS* representa uma espécie de satisfação objetiva. Por outro lado, a Avaliação Subjetiva reflete o sentimento pessoal de satisfação do profissional ao analisar uma solução quanto ao atendimento de seus anseios. Portanto, a avaliação concedida pelo tomador de decisão é, essencialmente, uma satisfação subjetiva.

O Quadro 6, apresenta um recorte da Tabela 6 mostrando apenas os resultados de *NS* atingidos pela solução final, aceita por cada um dos participantes, bem como a avaliação subjetiva que os mesmos forneceram para tal solução.

Quadro 6 – Recorte da Tabela 6 mostrando resultados de *NS* e Avaliação Subjetiva

Participantes	**#1**	**#2**	**#3**	**#4**	**#5**	**#6**	**#7**	**#8**	**#9**	**#10**	**Média**
NS (*Si*)	1,00	1,00	0,90	0,88	1,00	0,90	0,76	1,00	0,86	1,00	$0{,}93 \pm 0{,}08$
Avaliação Subjetiva (*Si*)	0,93	0,87	0,88	0,75	0,93	0,84	0,75	0,88	0,50	0,87	$0{,}82 \pm 0{,}13$

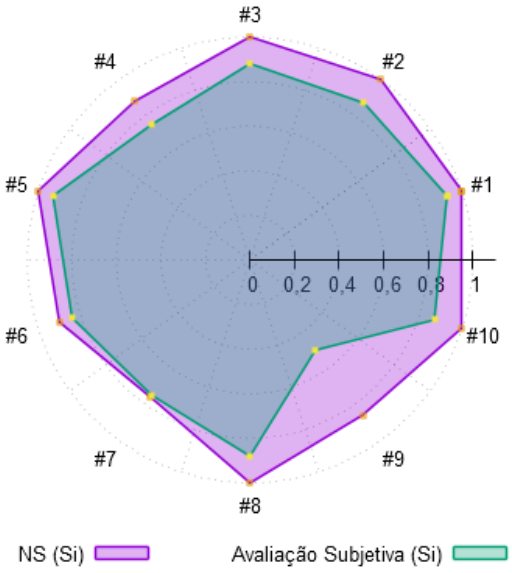
Fonte: Elaborado pelo autor.

Observa-se que as avaliações subjetivas concedidas pelos participantes #1 e #2 chegaram a 93% do valor de *NS* para a solução de cada um deles. Porém, as avaliações subjetivas e objetivas que mais se aproximaram foram as do participante #7, com valores de 0,746 e 0,760. De forma geral, a maioria dos valores da satisfação subjetiva, ficaram a menos 0,14 dos valores de *NS*, considerando apenas a solução final do teste de cada participante. Novamente, o candidato #9 representou a exceção. Considerando os 10 participantes, a diferenças foi em média de 0,11.

Auxiliando na resposta da **QP6**: *"Considerando uma solução interativa, qual a relação entre a satisfação do tomador de decisão e o percentual de importância das preferências atendidas?"*, a Figura 19 mostra a relação entre ambas as métricas através de um gráfico.

Pelo que se constata, pode-se afirmar que o sentimento de satisfação do tomador de decisão diante de uma solução *Si* gerada pela abordagem, com ajuda do mesmo, na maioria das vezes é próximo, porém menor do que a satisfação objetiva, verificada através da importância das preferências satisfeitas por *Si*. Como o *NS* mensura apenas os aspectos introduzidos pela modelagem das preferências, tal inferioridade sugere que podem existir outros aspectos os quais exercem

Figura 19 – Relação entre Nível de Satisfação (NS) e Avaliação Subjetiva



Fonte: Elaborado pelo autor.

influência sobre satisfação subjetiva do tomador de decisão. Além disso, como o conjunto de preferências é limitado aos tipos de preferências modelados, em algum momento o usuário pode não ter conseguido expressar alguma opinião e, por consequência, ficado momentaneamente insatisfeito, o que obviamente poderia afetar o sentimento final de satisfação do participante.

5.4 AMEAÇAS À VALIDADE DOS EXPERIMENTOS

O processo de experimentação é uma atividade de pesquisa bastante complexa e, por consequência, altamente sensível a falhas. Por isso, é fundamental o conhecimento sobre os aspectos que podem de algum forma ameaçar a validade do estudo empírico e afetar as conclusões deste advindas. Nesse sentido, Wohlin *et al.* (2000) descrevem quatro categorias relativas à validade de experimentos no contexto da Engenharia de Software: Validade Interna, Validade Externa, Validade de Construção e Validade de Conclusão. De forma mais específica, Barros e Dias-Neto (2011a) apresentam um estudo sobre os fatores que podem ameaçar a validade de experimentos conduzido em SBSE.

Com relação à Validade Interna, pode-se destacar que apesar ter sido realizado um teste preliminar para definição das configurações dos parâmetros utilizados no Algoritmo Genético, outros valores para tais parâmetros poderiam produzir melhores resultados para alguma instância em específico. Não obstante, toda configuração aplicada foi descrita a fim de facilitar a replicação do experimento. Ao longo do texto não são apresentados os detalhes de implementação

das estruturas algorítmicas utilizadas, contudo, como forma de garantir transparência com relação a artifícios que pudessem favorecer a produção de resultados inconsistente, o código-fonte encontra-se totalmente disponível no material de suporte da pesquisa, conforme recomenda Johnson (2002). Além disso, das quatro instâncias utilizadas, duas foram geradas de forma aleatória e mesmo as outras duas, baseadas em dados reais, tiveram informações adicionadas aleatoriamente, ou seja, os dados testados não são completamente fidedignos a um ambiente real de desenvolvimento. No que se refere ao teste com humanos, os participantes podem ter sofrido com o chamado efeito *Hawthorne* (MCCAMBRIDGE; WITTON; ELBOURNE, 2014), descrito na literatura como a característica que os seres humanos possuem em mudar o comportamento ao saberem que estão sob avaliação.

Relacionado à Validade Externa dos experimentos, ou seja, à capacidade de generalização dos resultados alcançados, ressaltar-se que as duas instâncias baseadas em dados reais, possuíam pequena quantidade de requisitos. Assim, os resultados podem ser diferentes para uma instância de larga escala, por exemplo. Essa foi a justificativa para a geração de mais duas instâncias artificias contendo mais requisitos. Ainda assim, outras instâncias reais e com maior quantidade de requisitos ofereceriam maior generalidade aos resultados. Além disso, a quantidade de participantes do experimento com humanos pode não ser suficiente para representar os mais variados perfis de profissionais que atuam no planejamento de *releases*. Para tentar atenuar esse problema, foram selecionados participantes com diferentes perfis de atuação, porém com qualificação acadêmica e "média" ou "alta" experiência em desenvolvimento de software.

A Validade de Construção refere-se às relações entre teoria e observação, garantindo que o tratamento reflete a construção da causa e que o resultado reflete a construção do efeito. (BARROS; DIAS-NETO, 2011a). Nesse caso, o presente trabalhou não apresentou, por exemplo, avaliações relativas ao consumo de tempo do algoritmo de busca em função das características da instância. Em razão da formulação elaborada, as métricas utilizadas nas análises foram propostas no próprio trabalho ou aproveitadas de outras pesquisas que já haviam explorado a otimização interativa em contextos similares. Independente disso, todas as métricas foram claramente definidas, discutidas e eventualmente exemplificadas. Além disso, as modelagens dos aspectos próprios do Planejamento de *Releases* e da opinião do usuário, são simplificações do mundo real. No caso do experimento com profissionais, os participantes receberam explicações sobre o contexto do teste, porém não foram informados a cerca das hipóteses estudadas para não

influenciar o comportamento dos mesmos.

Finalmente, de acordo com Travassos, Gurov e Amaral (2002) a validade de conclusão diz respeito à habilidade de se chegar a uma conclusão correta a respeito dos relacionamentos entre o tratamento e o resultado do experimento. Nesse sentido, o uso da meta-heurística Algoritmo Genético, por possuir um caráter estocástico, produz resultados distintos para execuções diferentes para uma mesma configuração. Por esse motivo, cada combinação testada foi executada por 30 vezes, colhendo-se os valores para cada métrica analisada, conforme recomendação de Arcuri e Briand (2011). Além disso, a maioria das constatações obtidas foram apoiadas por testes estatísticos. Como não foram encontradas outras abordagens que tratassem o problema do Planejamento de *Releases* com foco na modelagem da opinião do tomador de decisão, a linha base de comparação foi uma versão não-interativa implementada em consonância com modelos já estabelecidos na literatura.

5.5 CONCLUSÕES DO CAPÍTULO

Neste Capítulo foi apresentado o estudo empírico conduzido para avaliação da abordagem proposta. Tal experimentação foi norteada por seis questões de pesquisa.

Durante os experimentos, foram utilizadas instâncias baseadas em dados reais e artificiais, sendo também gerado um conjunto de preferências para cada uma delas. Para avaliação exaustiva da abordagem, foi realizado um experimento automático com inúmeras combinações de configurações. As possibilidades envolveram todas as instâncias, variações de nível de influência da Base de Preferências e da quantidade de preferências consideradas durante o processo de busca. Para avaliar o comportamento da técnica proposta, quando usada por um profissional, 10 participantes foram selecionados para realizar o planejamento das *releases* para a instância "Processador de Textos" utilizando uma ferramenta desenvolvida para este fim.

Através dos resultados e análises, ficou demonstrado que a abordagem é capaz de incluir as preferências humanas durante a busca, priorizando aquelas mais relevantes para o tomador de decisão e tendo como consequência, uma certa perda no *score* das soluções. Além disso, a abordagem mostrou-se consistente e eficaz quando utilizada por profissionais experiente na área de desenvolvimento.

Por fim, foram discutidos aspectos que podem constituírem-se como ameaças à validade dos experimentos bem como as eventuais medidas adotadas para mitigar potenciais prejuízos.

6 CONSIDERAÇÕES FINAIS

O planejamento de *releases* é a atividade central do desenvolvimento de software incremental e, por envolver diversos aspectos, muitas vezes conflitantes, a definição de quais requisitos devem ser implementados em cada versão é uma tarefa complexa cognitiva e computacionalmente. O problema de planejar a sequência de funcionalidades a serem liberadas para os clientes, é conhecido como problema do Planejamento de *Releases* e tem sido bastante explorado em *SBSE*. Entretanto, até o presente momento não existem trabalhos com foco nas preferências que o tomador de decisão pode possuir durante o processo de geração das soluções.

Nesse contexto, o presente trabalho teve como principal objetivo propor e implementar uma abordagem interativa usando uma Base de Preferências de modo a aliar a expertise humana ao poder computacional das técnicas de busca.

Através do estudo empírico conduzido, constatou-se que a abordagem proposta é de fato sensível a interação humana, gerando soluções que satisfazem até 100% das preferências do usuário, e quando isso não for possível, as mais importantes são priorizadas. Contudo, verificou-se também que a inclusão da opinião humana no processo de geração das soluções acarreta perda de até 11% na otimização de fatores próprios do Planejamento de *Releases*. Além disso, viu-se que a probabilidade de atendimento das preferências é inversa à quantidade de preferências presentes na Base de Preferências. Finalmente, averiguou-se que ao se incluir o tomador de decisão no processo de construção das soluções, sua satisfação aumenta em média de 44% para 82% em comparação às soluções não interativas.

Em resumo, a presente pesquisa mostrou que é possível tratar o Planejamento de *Releases* como um problema de Otimização Interativa, obtendo alta satisfação do tomador de decisão sem grandes prejuízos aos critérios objetivos do problema.

6.1 CONTRIBUIÇÕES

Além de crescimento intelectual para o autor, outras contribuições foram produzidas durante a execução desta pesquisa, sendo as principais delas listadas a seguir:

a) **Modelagem das preferências do tomador de decisão:** neste trabalho foi estabelecido um modelo flexível e extensível para definição de tipos de preferências do tomador de decisão para o Planejamento de *Releases*. A partir desse modelo, foram formalizados 8 tipos de preferências relacionadas à alocação de requisitos;

b) **Formulação interativa para Planejamento de *Releases*:** uma vez existindo a modelagem das preferências, foi proposto um componente interativo à formulação do Planejamento de *Releases*, interpretando-o não só como um problema de busca, mas como um problema de Otimização Interativa;

c) **Implementação de ferramenta:** como um dos principais desafios de muitas abordagens de SBSE é a aplicação prática das técnicas propostas, neste trabalho desenvolveu-se uma ferramenta que implementa os princípios da abordagem, possibilitando o carregamento de instâncias, a visualização de soluções e o processo de interação do tomador de decisão;

d) **Publicação e Prêmio:** um artigo com a proposta inicial e resultados preliminares desta pesquisa foi submetido e aceito para publicação no *7th Symposium on Search-Based Software Engineering*, ocorrido em Bérgamo na Itália, entre os dias 5 e 7 de setembro de 2015. Em função do ineditismo tanto da formulação interativa elaborada para o Planejamento de *Releases*, quanto da modelagem das preferências do tomador de decisão, dos resultados preliminares, e do reconhecimento da comunidade como pesquisa promissora, o artigo nomeado *"Interactive Software Release Planning with Preferences Base"* foi premiado como melhor artigo na trilha de estudantes sendo certificado como *"Best Student Paper with industry-relevant SBSE results"*.

6.2 LIMITAÇÕES

A partir do que foi projetado como objetivos e dos resultados obtidos, considera-se esta pesquisa exitosa, todavia, é reconhecido que alguns aspectos podem ser compreendidos como pontos limitadores deste trabalho, são eles:

a) Abstração de alguns aspectos que outras abordagens em SBSE consideram para o Planejamento de *Releases*;

b) Limitada quantidade de tipos de preferências do tomador de decisão;

c) Inexistências de mecanismo de identificação de preferências conflitantes e redundantes;

d) Inexistências de validação da técnica proposta em um ambiente real de desenvolvimento de software;

e) Uso de apenas uma técnica de otimização para geração de soluções.

6.3 TRABALHOS FUTUROS

Os resultados alcançados e as conclusões obtidas demonstraram a viabilidade da abordagem proposta mas também apontam para necessidade de evolução da pesquisa. Por isso, segue abaixo uma lista de trabalhos futuros que podem complementar esta pesquisa:

a) Ampliar o conjunto de tipos de preferências;
b) Adicionar à formulação do Planejamento de *Releases* alguns aspectos que foram abstraídos na atual proposta, tais como, priorização e similaridade de requisitos;
c) Considerar outras restrições, como, por exemplo, tempo de projeto e interdependência de custo e valor requisitos;
d) Testar outras técnicas de busca como mecanismo de otimização;
e) Projetar e desenvolver um mecanismo para detecção de conflitos lógicos e redundância de sentido entre as preferências;
f) Implementar melhorias na ferramenta, tendo como direcionamento as sugestões apontadas pelos participantes do estudo empírico;
g) Evoluir a formulação proposta para uma versão multiobjetivo, utilizando as preferências como objetivo ou usando a base para auxiliar a escolha de uma solução da Frente de Pareto.

REFERÊNCIAS

ABURAS, A.; GROCE, A. An improved memetic algorithm with method dependence relations (mamdr). In: **Proceedings of the 14th International Conference on Quality Software (QSIC '14)**. Allen, TX, USA: IEEE, 2014. p. 11–20.

AL-EMRAN, A.; PFAHL, D. Operational planning, re-planning and risk analysis for software releases. In: **Product-Focused Software Process Improvement**. [S.l.]: Springer, 2007. p. 315–329.

ALI, S.; IQBAL, M. Z. Improved heuristics for solving ocl constraints using search algorithms. In: **Proceedings of the 2014 Conference on Genetic and Evolutionary Computation (GECCO '14)**. Vancouver, Canada: ACM, 2014. p. 1231–1238.

ALJAWAWDEH, H. J.; SIMONS, C. L.; ODEH, M. Metaheuristic design pattern: Preference. In: **Proceedings of the Companion Publication of the 2015 on Genetic and Evolutionary Computation Conference**. New York, NY, USA: ACM, 2015. (GECCO Companion '15), p. 1257–1260. ISBN 978-1-4503-3488-4. Disponível em: <http://doi.acm.org/10.1145/2739482.2768498>.

ALJAWAWDEH, H. J.; SIMONS, C. L.; ODEH, M. Metaheuristic design pattern: Preference. In: ACM. **Proceedings of the Companion Publication of the 2015 on Genetic and Evolutionary Computation Conference**. [S.l.], 2015. p. 1257–1260.

AMANDEEP, A.; RUHE, G.; STANFORD, M. Intelligent support for software release planning. In: **Proceedings of the 5th International Conference on Product Focused Software Process Improvement (PROFES '04)**. Kansai Science City, Japan: Springer, 2004. v. 3009, p. 248–262.

ANDRADE, E. L. de. **Introdução à pesquisa operacional: métodos e modelos para a análise de decisão**. [S.l.]: Livros Técnicos e Científicos, 1998.

ARAúJO, A. A.; PAIXãO, M. H. E. Machine learning for user modeling in an interactive genetic algorithm for the next release problem. In: **Proceedings of the 6th International Symposium on Search-Based Software Engineering (SSBSE '14)**. Fortaleza, Brazil: Springer, 2014. v. 8636, p. 228–233.

ARAúJO, A. A. P. **Uma Arquitetura Utilizando Algoritmo Genético Interativo e Aprendizado de Máquina Aplicado ao Problema Do Próximo *Release***. Dissertação (Mestrado) — Universidade Estadual do Ceará, Fortaleza, 2014.

ARCURI, A.; BRIAND, L. A practical guide for using statistical tests to assess randomized algorithms in software engineering. In: **Proceedings of the 33rd International Conference on Software Engineering**. New York, NY, USA: ACM, 2011. (ICSE '11), p. 1–10. ISBN 978-1-4503-0445-0.

ASSUNçãO, W. K. G.; COLANZI, T. E.; VERGILIO, S. R.; POZO, A. A multi-objective optimization approach for the integration and test order problem. **Information Sciences**, v. 267, p. 119–139, May 2014.

BAGNALL, A. J.; RAYWARD-SMITH, V. J.; WHITTLEY, I. M. The next release problem. **Information and software technology**, Elsevier, v. 43, n. 14, p. 883–890, 2001.

BAKER, P.; HARMAN, M.; STEINHöFEL, K.; SKALIOTIS, A. Search based approaches to component selection and prioritization for the next release problem. In: **Proceedings of the 22nd IEEE International Conference on Software Maintenance (ICSM '06)**. Philadelphia, Pennsylvania: IEEE, 2006. p. 176–185.

BARROS, M. d. O.; DIAS-NETO, A. C. **Threats to Validity in Search-based Software Engineering Empirical Studies**. [S.l.], 2011. v. 5, n. 0006/2011.

BARROS, M. de O.; DIAS-NETO, A. C. A survey of empirical investigations on ssbse papers. In: **Proceedings of the 3rd International Symposium on Search Based Software Engineering (SSBSE '11)**. Szeged, Hungary: Springer, 2011. v. 6956, p. 268–268.

BAVOTA, G.; CARNEVALE, F.; LUCIA, A. D.; PENTA, M. D.; OLIVETO, R. Putting the developer in-the-loop: An interactive ga for software re-modularization. In: **Proceedings of the 4th International Symposium on Search Based Software Engineering (SSBSE '12)**. Riva del Garda, Italy: Springer, 2012. v. 7515, p. 75–89.

BRASIL, M. M. A. **Planejamento de releases de software através da aplicação de técnicas de busca multiobjetivas**. Dissertação (Mestrado) — Universidade Estadual do Ceará, Fortaleza, 2011.

BRASIL, M. M. A.; SILVA, T. G. N. da; FREITAS, F. G. de; SOUZA, J. T. de; CORTéS, M. I. A multiobjective optimization approach to the software release planning with undefined number of releases and interdependent requirements. In: **Proceedings of the 13th International Conference on Enterprise Information Systems (ICEIS '11)**. Beijing, China: Springer, 2011.

CARLSHAMRE, P.; SANDAHL, K.; LINDVALL, M.; REGNELL, B. *et al.* An industrial survey of requirements interdependencies in software product release planning. In: IEEE. **in Proceedings of Fifth IEEE International Symposium on Requirements Engineering (2001)**. [S.l.], 2001. p. 84–91.

COLARES, F.; SOUZA, J. T. de; CARMO, R. A. F. do; PáDUA, C.; MATEUS, G. R. A new approach to the software release planning. In: **Proceedings of the 23rd Brazilian Symposium on Software Engineering (SBES '09)**. Fortaleza, Ceará, Brazil: IEEE, 2009. p. 207–215.

CUNHA, A. G.; TAKAHASHI, R.; ANTUNES, C. A. H. d. C. **Manual de computação evolutiva e metaheurística**. Coimbra: Imprensa da Universidade de Coimbra, 2012. ISBN 978-989-26-0583-8 (PDF). Disponível em: <https://digitalis.uc.pt/handle/10316.2/5655>.

DARWIN, C.; BYNUM, W. F. **The origin of species by means of natural selection: or, the preservation of favored races in the struggle for life**. [S.l.]: AL Burt, 2009.

DEB, K. Advances in evolutionary multi-objective optimization. In: **Search Based Software Engineering**. [S.l.]: Springer, 2012. p. 1–26.

DUENAS, A.; TÜTÜNCÜ, G. Y.; CHILCOTT, J. B. A genetic algorithm approach to the nurse scheduling problem with fuzzy preferences. **IMA Journal of Management Mathematics**, IMA, v. 20, n. 4, p. 369–383, 2009.

DURILLO, J. J.; NEBRO, A. J. jmetal: A java framework for multi-objective optimization. **Advances in Engineering Software**, v. 42, p. 760–771, 2011. ISSN 0965-9978. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0965997811001219>.

EIBEN, A. E.; SMITH, J. E. **Introduction to evolutionary computing**. [S.l.]: Springer Science & Business Media, 2003.

FERREIRA, T. do N.; SOUZA, J. T. de. An aco approach for the next release problem with dependency among requirements. In: **Proceedings of the 3rd Brazilian Workshop on Search-Based Software Engineering (WESB '12)**. Natal, RN, Brazil: [s.n.], 2012.

FERREIRA, T. N. **Abordagens Interativas Usando Algoritmo de Otimização por Colônia de Formiga para O Problema do Próximo *Release***. Dissertação (Mestrado) — Universidade Estadual do Ceará, Fortaleza, 2015.

FIGUEIRA, J.; MOUSSEAU, V.; ROY, B. Electre methods. In: **Multiple criteria decision analysis: State of the art surveys**. [S.l.]: Springer, 2005. p. 133–153.

FISHER, M. Interactive optimization. **Annals of Operations Research**, Springer, v. 5, n. 1-4, p. 541–556, 1986.

FUCHSHUBER, R.; BARROS, M. de O. Improving heuristics for the next release problem through landscape visualization. In: **Proceedings of the 6th International Symposium on Search-Based Software Engineering (SSBSE '14)**. Fortaleza, Brazil: Springer, 2014. v. 8636, p. 222–227.

GENDREAU, M.; POTVIN, J.-Y. **Handbook of metaheuristics**. [S.l.]: Springer, 2010.

GREER, D.; RUHE, G. Software release planning: An evolutionary and iterative approach. **Information & Software Technology**, v. 46, n. 4, p. 243–253, March 2004.

GUIZZO, G.; COLANZI, T. E.; VERGILIO, S. R. A pattern-driven mutation operator for search-based product line architecture design. In: **Proceedings of the 6th International Symposium on Search-Based Software Engineering (SSBSE '14)**. Fortaleza, Brazil: Springer, 2014. v. 8636, p. 77–91.

HARMAN, M.; CLARK, J. A. Metrics are fitness functions too. In: **Proceedings of the 10th IEEE International Symposium on Software Metrics (METRICS '04)**. Chicago, USA: IEEE, 2004. p. 58–69.

HARMAN, M.; JONES, B. F. Search-based software engineering. **Information and Software Technology**, Elsevier, v. 43, n. 14, p. 833–839, 2001.

HARMAN, M.; MCMINN, P.; SOUZA, J. T. de; YOO, S. Search based software engineering: Techniques, taxonomy, tutorial. **Empirical Software Engineering and Verification**, v. 7007, p. 1–59, 2012.

HART, J.; SHEPPERD, M. J. Evolving software with multiple outputs and multiple populations. In: **Proceedings of the 2002 Conference on Genetic and Evolutionary Computation (GECCO '02)**. New York, USA: Morgan Kaufmann Publishers, 2002. p. 223–227. Disponível em: <http://decgradschool.bournemouth.ac.uk/ESERG/Technical_Reports/TR02-06/TR02-06.pdf>.

HOLLAND, J. H. **Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence**. [S.l.]: USA: University of Michigan, 1975.

JIANG, T.; GOLD, N.; HARMAN, M.; LI, Z. Locating dependence structures using search-based slicing. **Information and Software Technology**, v. 50, n. 12, p. 1189–1209, November 2007.

JOHNSON, D. S. A theoretician's guide to the experimental analysis of algorithms. **Data structures, near neighbor searches, and methodology: fifth and sixth DIMACS implementation challenges**, v. 59, p. 215–250, 2002.

JONG, K. A. D. Analysis of the behavior of a class of genetic adaptive systems. 1975.

KARIM, M. R.; RUHE, G. Bi-objective genetic search for release planning in support of themes. In: **Proceedings of the 6th International Symposium on Search-Based Software Engineering (SSBSE '14)**. Fortaleza, Brazil: Springer, 2014. v. 8636, p. 123–137.

KUKUNAS, J.; CUPPER, R. D.; KAPFHAMMER, G. M. A genetic algorithm to improve linux kernel performance on resource-constrained devices. In: **Proceedings of the 12th Annual Conference on Genetic and Evolutionary Computation (GECCO '10)**. Portland, Oregon, USA: ACM, 2010. p. 2095–2096.

LACERDA, E. G. de; CARVALHO, A. de. Introdução aos algoritmos genéticos. **Sistemas inteligentes: aplicaçoes a recursos hıdricos e ciências ambientais**, v. 1, p. 99–148, 1999.

LINDEN, R. **Algoritmos genéticos (3a ediçao)**. [S.l.]: Ciência Moderna, 2012.

MAIA, C. L. B. **Uma Abordagem Integrada, Interativa e Multi-Objetiva para os Problemas de Seleção, Priorização e Alocação de Casos de Teste**. Dissertação (Mestrado) — MACC, Mestrado Acadêmico em Ciência da Computação, Universidade Estadual do Ceará, 2011.

MARCULESCU, B.; FELDT, R.; TORKAR, R. A concept for an interactive search-based software testing system. In: **Proceedings of the 4th International Symposium on Search Based Software Engineering (SSBSE '12)**. Riva del Garda, Italy: Springer, 2012. v. 7515, p. 273–278.

MCCAMBRIDGE, J.; WITTON, J.; ELBOURNE, D. R. Systematic review of the hawthorne effect: new concepts are needed to study research participation effects. **Journal of clinical epidemiology**, Elsevier, v. 67, n. 3, p. 267–277, 2014.

MIETTINEN, K. **Nonlinear multiobjective optimization**. [S.l.]: Springer Science & Business Media, 2012.

NAKAGAWA, S.; CUTHILL, I. C. Effect size, confidence interval and statistical significance: a practical guide for biologists. **Biological Reviews**, Wiley Online Library, v. 82, n. 4, p. 591–605, 2007.

NGO-THE, A.; RUHE, G. A systematic approach for solving the wicked problem of software release planning. **Soft Computing - A Fusion of Foundations, Methodologies and Applications**, v. 12, n. 1, p. 95–108, August 2008.

PITANGUEIRA, A. M. Incorporating preferences from multiple stakeholders in software requirements selection an interactive search-based approach. In: IEEE. **Requirements Engineering Conference (RE), 2015 IEEE 23rd International**. [S.l.], 2015. p. 382–387.

R Core Team. **R: A Language and Environment for Statistical Computing**. Vienna, Austria, 2015. Disponível em: <https://www.R-project.org>.

RAMREZ, A.; ROMERO, J. R.; VENTURA, S. A novel component identification approach using evolutionary programming. In: **Proceeding of The 15th Annual Conference Companion on Genetic and Evolutionary Computation (GECCO '13)**. Amsterdam, The Netherlands: ACM, 2013. p. 209–210.

RUHE, G. **Product release planning: methods, tools and applications**. Boca Raton, FL, USA: CRC Press, 2010.

RUHE, G.; NGO-THE, A. Hybrid intelligence in software release planning. **International Journal of Hybrid Intelligent Systems**, v. 1, n. 1-2, p. 99–110, April 2004. Disponível em: <http://iospress.metapress.com/content/7ad6fldflt80e9vg/>.

RUHE, G.; SALIU, M. O. The art and science of software release planning. **Software, IEEE**, IEEE, v. 22, n. 6, p. 47–53, 2005.

RUIZ, A. B.; SABORIDO, R.; LUQUE, M. A preference-based evolutionary algorithm for multiobjective optimization: the weighting achievement scalarizing function genetic algorithm. **Journal of Global Optimization**, Springer, v. 62, n. 1, p. 101–129, 2014.

SALIU, O.; RUHE, G. Supporting software release planning decisions for evolving systems. In: IEEE. **Software Engineering Workshop, 2005. 29th Annual IEEE/NASA**. [S.l.], 2005. p. 14–26.

SHACKELFORD, M.; SIMONS, C. L. Metaheuristic design pattern: interactive solution presentation. In: ACM. **Proceedings of the 2014 conference companion on Genetic and evolutionary computation companion**. [S.l.], 2014. p. 1431–1434.

SIMONS, C. L.; SMITH, J.; WHITE, P. Interactive ant colony optimization (iaco) for early lifecycle software design. **Swarm Intelligence**, v. 8, n. 2, p. 139–157, June 2014.

SOMMERVILLE, I. **Software Engineering**. [S.l.]: Addison Wesley, 2011.

SOUZA, J. T. D.; MAIA, C. L.; FREITAS, F. G. D.; COUTINHO, D. P. The human competitiveness of search based software engineering. In: IEEE. **Search Based Software Engineering (SSBSE), 2010 Second International Symposium on**. [S.l.], 2010. p. 143–152.

SOUZA, J. T. de; MAIA, C. L. B.; FERREIRA, T.; CARMO, R. A. F. do; BRASIL, M. An ant colony optimization approach to the software release planning with dependent requirements. In: **Proceedings of the 3rd International Symposium on Search Based Software Engineering (SSBSE '11)**. Szeged, Hungary: Springer, 2011. v. 6956, p. 142–157.

SRINIVAS, M.; PATNAIK, L. M. Genetic algorithms: A survey. **Computer**, IEEE, v. 27, n. 6, p. 17–26, 1994.

TAKAGI, H. Interactive evolutionary computation: Fusion of the capabilities of evolutionary computation optimization and human evaluation. **Proceedings of the IEEE**, IEEE, v. 89, n. 9, p. 1275–1296, 2001.

THIELE, L.; MIETTINEN, K.; KORHONEN, P. J.; MOLINA, J. A preference-based evolutionary algorithm for multi-objective optimization. **Evolutionary Computation**, MIT Press, v. 17, n. 3, p. 411–436, 2009.

TONELLA, P.; SUSI, A.; PALMA, F. Using interactive ga for requirements prioritization. In: **Proceedings of the 2nd International Symposium on Search Based Software Engineering (SSBSE '10)**. Benevento, Italy: IEEE, 2010. p. 57–66.

TRAVASSOS, G. H.; GUROV, D.; AMARAL, E. **Introdução à engenharia de software experimental**. [S.l.]: UFRJ, 2002.

VARGHA, A.; DELANEY, H. D. A critique and improvement of the cl common language effect size statistics of mcgraw and wong. **Journal of Educational and Behavioral Statistics**, Sage Publications, v. 25, n. 2, p. 101–132, 2000.

WOHLIN, C.; RUNESON, P.; HöST, M.; OHLSSON, M. C.; REGNELL, B.; WESSLÉN, A. **Experimentation in Software Engineering: An Introduction**. Norwell, MA, USA: Kluwer Academic Publishers, 2000. ISBN 0-7923-8682-5.

ZHANG, M. H. Y.; MANSOURI, A. **The SBSE Repository: A repository and analysis of authors and research articles on Search Based Software Engineering**. 2015. <http:/http://crestweb.cs.ucl.ac.uk/resources/sbse_repository>.

ZHANG, Y.; HARMAN, M.; MANSOURI, S. A. The multi-objective next release problem. In: ACM. **Proceedings of the 9th annual conference on Genetic and evolutionary computation**. [S.l.], 2007. p. 1129–1137.

APÊNDICES

APÊNDICE A – Termo de Consentimento

TERMO DE CONSENTIMENTO LIVRE E ESCLARECIDO

Eu,_____,RG_____, CPF_____, abaixo assinado, autorizo a Universidade Estadual do Ceará (UECE), por intermédio dos alunos e pesquisadores do GOES (Grupo de Otimização em Engenharia de Software), Altino Dantas Basílio Neto, Allysson Allex de Paula Araújo, Italo Yeltsin Medeiros Bruno devidamente assistidos pelo seu orientador Prof. Dr. Jerffeson Teixeira de Souza, consinto a desenvolver a pesquisa abaixo descrita:

1) Título: "**An Interactive Approach to Software Release Planning supported by a Dynamic Preferences Base**"

2) Objetivo: Avaliar o impacto das preferências humanas durante um processo automatizado de seleção de requisitos.

3) Descrição de procedimentos: Aplicação de questionários e realização de testes em laboratório.

4) Desconfortos e riscos esperados: Nenhum. Fui devidamente informado dos riscos acima descritos e de qualquer risco não descrito, não previsível, porém os que possam ocorrer em decorrência da pesquisa serão de inteira responsabilidade dos pesquisadores.

5) Benefícios esperados: Disponibilizar, com finalidade acadêmica e/ou comercial, uma ferramenta de seleção de requisitos que esteja apta a incorporar a subjetividade humana durante a automatização do processo de definição do Planejamento de Releases de Software.

6) Informações: Os participantes têm a garantia que receberão respostas a qualquer pergunta e esclarecimento de qualquer dúvida quanto aos assuntos relacionados à pesquisa. Também os pesquisadores supracitados assumem o compromisso de proporcionar informações atualizadas obtidas durante a realização do estudo.

7) Retirada do consentimento: O voluntário tem a liberdade de retirar seu consentimento a qualquer momento e deixar de participar do estudo, não acarretando nenhum ônus ou dano ao voluntário.

8) Aspecto Legal: Elaborado de acordo com as diretrizes e normas regulamentadas de pesquisa envolvendo seres humanos, atendendo à Resolução no 196, de 10 de outubro de 1996, do Conselho Nacional de Saúde do Ministério de Saúde - Brasília – DF.

9) Confiabilidade: Os voluntários terão direito à privacidade. A identidade (nomes e sobrenomes) do participante não será divulgada, porém os voluntários assinarão o termo de consentimento para que os resultados obtidos possam ser apresentados em eventos científicos ou publicações.

10) Quanto à indenização: Não há danos previsíveis decorrentes da pesquisa; mesmo assim fica prevista indenização, caso se faça necessário.

Fortaleza (CE), _____de _____de 2016.

_____

Assinatura do Voluntário

APÊNDICE B – Questionário de coleta de dados utilizado no levantamento de perfil dos profissionais participantes do experimento.

QUESTIONÁRIO DE COLETA DE DADOS

Mestrado Acadêmico em Ciência da Computação
Universidade Estadual do Ceará

Q1. Qual a sua formação acadêmica? Marque a opção que representa a sua última formação.

- Ensino Médio Concluído
- Graduação em Andamento
- Graduação Concluída
- Pós-Graduação em Andamento
- Pós-Graduação Concluída

Q2. Escreva abaixo sua atual ocupação profissional:

Q3. Quanto tempo de experiência você possui na área de TI?

Q4. Como você classificaria a sua experiência em desenvolvimento de software:

- Alta
- Média
- Baixa

Q5. Como você classificaria seu conhecimento com relação a seleção de requisitos:

- Alto
- Médio
- Baixo

Q6. Você utiliza ou já utilizou algum método de seleção **automática** de requisitos no seu ambiente de trabalho? Por favor, marque uma das alternativas e complemente se necessário.

- Sim, usei a(o)_____.
- Não, nunca tive conhecimento de métodos com essa característica.
- Não, os métodos automáticos que conheço não são satisfatórios.
- Não, porque os métodos automáticos que conheço_____
  _____
  _____

Q7. Você utiliza ou já utilizou algum método de seleção **manual** de requisitos no seu ambiente de trabalho? Se sim, qual?

Obrigado!

APÊNDICE C – Questionário de *feedback* utilizado para permitir aos participantes expressarem suas impressões sobre o experimento com o qual contribuíram.

QUESTIONÁRIO DE FEEDBACK

Mestrado Acadêmico em Ciência da Computação
Universidade Estadual do Ceará

Q1. O que você achou das explicações antes do uso efetivo da ferramenta?
- Ótimas
- Boas
- Razoáveis
- Ruins
- Péssimas

Q2. No geral, quão eficaz você achou a experiência de auxiliar interativamente a resolução do Planejamento de Releases de Software? Por favor, marque a resposta mais apropriada.
- Muito eficaz
- Eficaz
- Indiferente
- Ineficaz
- Muito Ineficaz

Q3. O quão fácil foi expressar sua opinião através do conjunto de preferências disponíveis? Por favor, marque a resposta mais apropriada.
- Muito fácil
- Fácil
- Indiferente
- Difícil
- Muito difícil

Q4. Você sugeriria alguma outra forma de expressar sua opinião? Detalhe abaixo.

Q5. Como profissional atuando na área de engenharia de requisitos, você utilizaria essa abordagem interativa?
- Com certeza
- Provavelmente
- Possivelmente
- Dificilmente
- De forma alguma

Q6. Quanto a apresentação da solução final, a interface da ferramenta pode ser considerada?
- Muito eficiente
- Eficiente
- Indiferente
- Deficiente
- Muito deficiente

Q7. Você gostaria de sugerir alguma mudança na interface do sistema? Detalhe abaixo.

Obrigado!