

UNIVERSIDADE FEDERAL DE OURO PRETO

pRINS: uma matheurística para problemas binários

Thiago Macedo Gomes
Universidade Federal de Ouro Preto

Orientador: Marcone Jamilson Freitas Souza

Co-orientador: Haroldo Gambini Santos

Dissertação submetida ao Instituto de Ciências
Exatas e Biológicas da Universidade Federal de
Ouro Preto para obtenção do título de Mestre
em Ciência da Computação

Ouro Preto, 30 de Janeiro de 2014

pRINS: uma matheurística para problemas binários

Thiago Macedo Gomes
Universidade Federal de Ouro Preto

Orientador: Marcone Jamilson Freitas Souza

Co-orientador: Haroldo Gambini Santos



Dedico este trabalho aos meus pais Geraldo e Fátima, meus irmãos João e Lucas, minha tia Maria Auxiliadora. Todas pessoas de suma importância na minha vida.

pRINS: uma matheurística para problemas binários

Resumo

Uma importante técnica para resolver problemas de otimização é por meio de Programação Inteira Mista (MIP, do inglês *Mixed Integer Programming*). Uma formulação MIP de um problema envolve um conjunto de variáveis, um conjunto de restrições sobre estas variáveis, um conjunto de restrições de integralidade e uma função objetivo linear a otimizar. Aplicações em otimização inteira são encontradas em diversas áreas do conhecimento, incluindo-se roteamento de veículos, alocação de enfermeiros, programação de horários, entre outros. O uso de métodos heurísticos tem sido empregado na resolução de problemas MIP como uma forma de acelerar o processo de busca na árvore de *branching*. Este trabalho propõe uma adaptação da heurística MIP *Relaxation Induced Neighborhood Search* (RINS), a qual explora a idéia de fixar variáveis de mesmo valor na solução inteira e fracionária corrente. O método proposto, denominado pRINS, explora explicitamente técnicas de pré-processamento, procurando sistematicamente por um número ideal de fixações, visando a produzir sub-problemas de tamanho controlado. As variáveis a fixar são organizadas por meio de um vetor de prioridade, onde foram propostas três maneiras de escolha destas variáveis, cada uma delas dando origem a uma variante do método. Em seguida os problemas são criados e resolvidos de modo semelhante ao método *Variable Neighborhood Descent* até que um critério de parada seja satisfeito. Os resultados das variantes do método foram comparados com os do resolvidor COIN-OR CBC *stand-alone* e com o método RINS original. Pelos resultados obtidos, o método proposto se mostrou com desempenho superior a essas duas técnicas. **PALAVRAS CHAVE.** Matheurísticas, MIP Heurísticas, Otimização Inteira, RINS, Programação Matemática.

Declaração

Esta dissertação é resultado de meu próprio trabalho, exceto onde referência explícita é feita ao trabalho de outros, e não foi submetida para outra qualificação nesta nem em outra universidade.

Thiago Macedo Gomes

Agradecimentos

Em primeiro lugar agradeço a Deus, por sua força e presença em minha vida, mesmo que às vezes passe despercebido.

Em seguida, nestas poucas linhas agradeço a todos pela atenção, carinho, palavras de apoio, paciência e amizade daqueles que estiveram próximo a mim.

Aos meus pais, minha base, por sempre confiarem e acreditarem em mim, estando ao meu lado em todos os momentos e decisões.

Agradeço aos meus irmãos pelo companherismo e minha tia Maria Auxiliadora pelos conselhos.

Aos meus orientadores, Marccone Jamilson e Haroldo Gambini, pela dedicação, ensinamentos, conhecimento científico e profissional compartilhado.

Agradeço aos meus amigos Adriano, André, Bruno, Francisca, Jander, Kelly, Letiere, Luís Cláudio, Meire, Nayara, Sávio, Silvinha, Suelaine, Tomas, Yuri, Welli por compartilhar alegrias e tristezas. Aos amigos de 'república', Adelmo, Diego, Lucas e Sabrina, em que não dividimos apenas despesas, criamos laços de amizade.

Aos amigos e profissionais do NTI, do CEAD, do PPGCC/UFOP e todos aqueles que entraram em minha vida por outros meios.

Por fim, agradeço a todos os funcionários e professores do DECOM/UFOP pela formação técnica da qual tenho orgulho.

Muito Obrigado.

Sumário

Lista de Figuras	xiii
Lista de Tabelas	xv
Lista de Algoritmos	xvii
Nomenclatura	1
1 Introdução	3
1.1 Resolvedores	4
1.2 MIP Heurísticas	5
1.3 Objetivos	6
1.3.1 Objetivo Geral	6
1.3.2 Objetivos Específicos	6
1.4 Motivação	7
1.5 Organização do Texto	7
2 Heurísticas MIP	9
2.1 Heurísticas de Arredondamento	9
2.2 <i>Relaxation Induced Neighborhood Search</i>	12
2.3 <i>Local Branching</i>	14

2.4	<i>Feasibility Pump</i>	15
2.5	<i>Proximity Search</i>	18
2.6	<i>Evolutionary Algorithm</i>	20
2.7	<i>Relaxation Enforced Neighborhood Search</i>	21
3	pRINS: A heurística BP proposta	23
3.1	RINS	25
3.2	Busca Binária	26
3.3	Pré-processamento	27
3.4	Método pRINS Proposto	31
4	Experimentos Computacionais	39
4.1	Ambiente de Desenvolvimento	39
4.2	Caracterização das Instâncias	39
4.3	Experimentos e Resultados	42
4.4	Discussão dos Resultados	43
5	Considerações Finais e Trabalhos Futuros	55
A	Apêndices	59
A.1	Publicações	59
	Referências Bibliográficas	61

Lista de Figuras

3.1	Esquema Geral do método pRINS	24
3.2	Método RINS	25
3.3	Esquema do método searchSubBP	28
3.4	Pré-processamento	28
3.5	Esquema do método SubBP	29
3.6	Esquema da fase 1 da heurística pRINS	34
3.7	Esquema da fase 2 da heurística pRINS	34

Lista de Tabelas

4.1	Instâncias MIPLIB 3	40
4.2	Instâncias MIPLIB 2003	41
4.3	Instâncias MIPLIB 2010	41
4.4	Resultados CBC, RINS	45
4.5	Resultados pRINS1	46
4.6	Resultados pRINS2, com Torneio Binário	47
4.7	Resultados pRINS2, com Torneio Ternário	48
4.8	Resultados pRINS2, com Torneio Quartenário	49
4.9	Resultados pRINS2, com Torneio Quinquenário	50
4.10	Resultados pRINS3, '10%' Shift	51
4.11	Resultados pRINS3, '25%' Shift	52
4.12	Resultados pRINS3, '50%' Shift	53
4.13	Resultados Consolidados: Soma e média dos gaps	54

Lista de Algoritmos

2.1	Simple Rounding	10
2.2	Feasibility Pump	16
3.1	searchSubBP	27
3.2	subBP	29
3.3	randomBiasedConstruction	30
3.4	shiftedConstruction	31
3.5	pRINS1	33
3.6	pRINS2	36
3.7	pRINS3	37

“Binary search is to algorithms what a wheel is to mechanics: It is simple, elegant, and immensely important.”

— Udi Manber, Introduction to Algorithms

Nomenclatura

BP	<i>Binary Problem</i>
COIN-OR	<i>COmputational INfrastructure for Operations Research</i>
FP	<i>Feasibility Pump</i>
LB	<i>Local Branching</i>
LP	<i>Linear Programming</i>
MIP	<i>Mixed Integer Program</i> (Problema de Programação Inteira Mista)
PI	<i>Programação Inteira</i>
PS	<i>Proximity Search</i>
RINS	<i>Relaxation Induced Neighborhood Search</i>
RENS	<i>Relaxation Enforced Neighborhood Search</i>
VND	<i>Variable Neighborhood Descent</i>
VNS	<i>Variable Neighborhood Search</i> (Busca em Vizinhança Variável)

Capítulo 1

Introdução

A programação inteira mista (MIP, das iniciais em inglês *Mixed Integer Programming*) é uma das mais importantes técnicas para resolver problemas de otimização complexos. Uma formulação MIP de um problema envolve um conjunto de variáveis, um conjunto de restrições sobre essas variáveis, um conjunto de restrições de integralidade e uma função objetivo linear a otimizar.

Aplicações de otimização inteira são encontradas nas mais diversas áreas, entre as quais: alocação de enfermeiros, alocação de tripulações aéreas, rodízio de tripulações, programação de horários, planejamento da produção, roteamento de veículos, rede de distribuição de energia, dimensionamento de frota de veículos e gerenciamento de projetos.

De acordo com (Fampa and Maculan 2006), os primeiros estudos para a resolução de funções lineares sujeitas a restrições se deram em 1826 com o trabalho de Fourier (Fourier 1890) sobre sistemas lineares de inequações. No entanto, o primeiro algoritmo para a solução de problemas de programação linear (PPL), denominado método SIMPLEX, foi desenvolvido por George Dantiz em 1947 (Dantzig 1951). O conjunto de restrições sobre as variáveis e as regras de integralidade definem em cada problema um politopo, sendo que em seus vértices encontram-se as soluções viáveis candidatas à solução ótima. Desta forma, em alto nível, a ideia do algoritmo simplex é então tomar um vértice e seguir na vizinhança deste, movendo-se entre os vértices enquanto as soluções encontradas estiverem melhorando.

Os MIPs são tipicamente resolvidos por técnicas *branch-and-bound* ou *branch-and-cut*. O método *branch-and-bound* foi desenvolvido a partir do trabalho de (Land and

Doig 1960) e consiste na enumeração implícita de todas as soluções candidatas. Durante esse processo os candidatos considerados não promissores, ou mesmo inviáveis, são descartados e, conseqüentemente, não expandidos. Os cortes na árvore de busca são baseados em estimativas de limite superior e inferior e evitam expandir vértices filho não promissores.

Por outro lado, o método *branch-and-cut* é uma generalização do *branch-and-bound*. Após a resolução do problema linear (LP, das iniciais em inglês *Linear Programming*), caso a poda não tenha sido bem sucedida, tenta-se encontrar um corte violado. Se um ou mais cortes violados são encontrados, eles são adicionados à formulação do problema e o LP é resolvido novamente. Se cortes não são encontrados, a ramificação é feita tal como no método *branch-and-bound*.

Atualmente o uso de outra técnica, os métodos heurísticos, também têm sido empregada na resolução de MIPs. Podemos definir heurística como sendo uma técnica inspirada em processos empíricos que procura uma boa solução a um custo computacional aceitável, sem, no entanto, estar capacitada a garantir sua otimalidade, bem como garantir quão próximo está da solução ótima (Souza 2014). Dentre os métodos heurísticos destacamos: Busca Tabu (Glover 1986, Souza 2012), Algoritmos Genéticos (Reeves 1993, Goldberg 1989), *Simulated Annealing* (Kirkpatrick, Gelatt and Vecchi 1983), *Iterated Local Search* (Paquete 2012) e GRASP (Resende, Mateus and Silva 2012).

1.1 Resolvedores

Um resolvedor é um termo genérico para indicar um software matemático, eventualmente sob a forma de um programa de computador ou como uma biblioteca de software, que se destina à solução de problemas de otimização. A partir das descrições do problema de forma genérica, é então determinada a solução. Uma grande necessidade para um MIP é a disponibilidade de programas resolvedores de alta qualidade. Alguns desses são livres; sendo assim, o usuário pode modificá-los de acordo com suas necessidades e as características do problema que está tratando.

Encontram-se disponíveis no mercado diversos resolvedores, tanto comerciais quanto de código aberto, dos quais podemos citar: CPLEX (*IBM CPLEX Optimizer* 2014), COIN-OR-CBC (*COIN-OR Branch-and-Cut MIP Solver* 2014), Symphony (*SYMPHONY* 2014), MINTO (*MINTO - Mixed INTeger Optimizer* 2014), SCIP (*Solving Constraint In-*

teger Programs 2014), GLPK (*GLPK - GNU Project* 2014) e Lp Solve (*lp solve reference guide* 2014).

Dentre os resolvidores livres, tem destaque o COIN-OR-CBC, que apresenta bons resultados quando comparado aos demais de sua categoria (*Mixed Integer Linear Programming Benchmark* 2011, *Mixed Integer Linear Programming Benchmark* 2013). O projeto COIN-OR surgiu a partir de um consórcio de pesquisadores da indústria e academia dedicados ao desenvolvimento da pesquisa operacional, hoje pertencente a uma corporação não lucrativa: COIN-OR Foundation (Forrest and Lougee-Heimer 2005, Lougee-Heimer 2003). Ele é voltado a problemas lineares e inteiros, e o pacote inclui recursos como pré-processamento, planos de corte, heurísticas e estratégias de *branching*.

Como principais usuários espera-se que sejam desenvolvedores de aplicação de algoritmos *branch-and-cut* personalizados em C++ usando as bibliotecas do CBC. Apesar de ter sido inicialmente projetado para ser usado como uma biblioteca, ele inclui um resolvidor independente que pode ser chamado pela linha de comando. As instâncias podem ser informadas ao resolvidor nos formatos de arquivo .lp e .mps. Também possui funcionalidades que permitem execução em modo paralelo para o aproveitamento de computadores com vários núcleos de processamento.

1.2 MIP Heurísticas

As heurísticas se originaram das comunidades de Pesquisa Operacional e Inteligência Artificial. Durante as primeiras décadas de pesquisa em metaheurísticas, coexistiam diferentes técnicas que no entanto não interagiam. Isto pode ser justificado, pelo fato de que para muitos problemas os métodos criados atingiam o estado da arte, funcionando bem separadamente. Apenas quando estes algoritmos atingiram seu limite, os pesquisadores começaram a considerar possíveis combinações com o objetivo de explorar os benefícios da sinergia entre os métodos. Estas abordagens são comumente referidas como metaheurísticas híbridas (Blum, Puchingerb, Raidl and Roli. 2011).

De fato, a escolha de uma combinação adequada de algoritmos complementares resulta em métodos de alta performance para a resolução de muitos problemas de otimização. Em (Blum, Puchingerb, Raidl and Roli. 2011) é apresentado um survey sobre a hibridização de metaheurísticas com outros métodos de otimização para resolução de problemas de natureza combinatória. Também é importante que estes métodos combi-

nem características de diversificação e intensificação na busca de uma solução.

Metaheurísticas que são guiadas através da relaxação de problemas lineares frequentemente são encontradas na literatura. Uma maneira simples de fazer uso de uma solução relaxada ótima de um problema, é uma heurística que a partir desta derive uma solução inteira viável. Dependendo do problema tratado, isto pode ser conseguido por simples arredondamento, através de fixações ou por estratégias mais sofisticadas de reparo.

Para (Eckstein and Nediak 2007), a motivação do uso de heurísticas em MIP é que estas devem ajudar a encontrar boas soluções mais cedo. Assim podem ajudar a limitar a quantidade de memória utilizada para armazenar a árvore e acelera o *branch-and-bound* por meio de técnicas como fixação do custo reduzido. Também em implementações paralelas de *branch-and-bound*, boas soluções iniciais evitam a busca em regiões de baixa qualidade na árvore, e ao mesmo tempo, intensificam a busca em regiões promissoras.

Este trabalho tem seu foco na resolução de MIPs que possuem apenas variáveis booleanas, os chamados *Binary Problems* (BP). Propõe-se uma heurística BP, que combina o poderio dos métodos de programação matemática com a flexibilidade dos métodos heurísticos. Por combinar essas duas classes de métodos, ela se enquadra nos chamados algoritmos matheurísticos.

1.3 Objetivos

1.3.1 Objetivo Geral

Este trabalho tem como objetivo geral desenvolver um algoritmo matheurístico eficiente baseado na combinação de metaheurísticas com programação matemática, para resolver problemas binários em programação linear.

1.3.2 Objetivos Específicos

São os seguintes objetivos específicos:

- Revisar os principais métodos matheurísticos encontrados na literatura;
- Estudar e implementar métodos utilizando o resolvidor COIN-OR CBC;

- Resolver instâncias de diversos tamanhos e tipos de problema;
- Contribuir com a pesquisa em programação linear inteira.

1.4 Motivação

Diversos problemas de otimização combinatória, incluindo situações práticas em engenharia, comércio, negócios e ciência podem ser modelados como problemas de programação inteira mista. Muitos destes modelos, tais como o problema da mochila, caixeiro viajante, empacotamento, roteamento em redes são conhecidos como pertencentes à classe de problemas NP-difíceis. Desta forma, a complexidade e os recursos computacionais requeridos para a resolução destes problemas crescem exponencialmente com o tamanho das instâncias.

Mesmo com as sucessivas melhorias nos métodos exatos, tais como *branch-and-bound* e *branch-and-cut*, muitos MIPs ainda não são resolvidos em tempo aceitável. Em vista disso, o desenvolvimento de MIP heurísticas (as chamadas matheurísticas), tem atraído a atenção dos pesquisadores como alternativa e complemento às abordagens clássicas. Encontrando uma boa solução mais cedo, principalmente nos problemas maiores, estes métodos proveem os limites inferior e superior e, portanto, podem acelerar os métodos de enumeração.

1.5 Organização do Texto

O restante deste texto encontra-se organizado da seguinte forma:

No Capítulo 2 é apresentada uma revisão de Heurísticas de Programação Inteira que têm apresentado bons resultados.

O Capítulo 3 apresenta o método matheurístico proposto.

No Capítulo 4 as instâncias utilizadas no trabalho são apresentadas, bem como detalhados os experimentos e os resultados obtidos.

O Capítulo 5 conclui o trabalho e apresenta perspectivas de trabalhos futuros.

Capítulo 2

Heurísticas MIP

Neste capítulo é apresentada uma revisão das heurísticas de Programação Inteira (PI) que têm obtido bons resultados na literatura.

2.1 Heurísticas de Arredondamento

Este grupo de heurísticas tem como característica básica o arredondamento de variáveis fracionárias para valores inteiros.

Consideremos um problema inteiro misto genérico da seguinte forma:

$$\min c^t x \tag{2.1}$$

$$s.t. Ax \leq b \tag{2.2}$$

$$x \geq 0 \tag{2.3}$$

$$x_j \in \mathbb{Z}, \forall j \in G \tag{2.4}$$

em que G é o conjunto das variáveis de decisão x_j , A é uma matriz $m \times n$ com elementos $a_{i,j}$, b um vetor m -dimensional e c é um vetor n -dimensional.

A heurística *Simple Rounding*, descrita por (Berthold 2006) e (Achterberg and Berthold 2007), verifica para cada variável inteira com valor fracionário, em um ponto viável, se esta é trivialmente arredondável para cima ou para baixo. Dado que todas as restrições

são do tipo “menor que”, uma variável x_j pode ser arredondada para baixo, se todos os coeficientes correspondentes à coluna j das restrições são não-negativos. De forma semelhante, se todos os coeficientes são não-positivos, uma variável pode ser trivialmente arredondada para cima.

Os autores também definem o conceito de *down-locks* e *up-locks*. O *down-lock* de uma variável x_j é o número de coeficientes $a_{i,j}$ que são negativos, assim como, o *up-lock* define o número de coeficientes $a_{i,j}$ positivos para esta variável. Uma variável que será arredondada não deve possuir nenhum lock. O pseudocódigo desta heurística é apresentado no Algoritmo 2.1.

Algoritmo 2.1: Simple Rounding

Input: $\bar{x}$ ponto LP-viável

```

1  $\bar{I} \leftarrow$  índice do conjunto de todas as variáveis fracionárias de  $\bar{x}$ ;
2 forall  $j \in \bar{I}$  do
3   if  $x_j$  é trivialmente arredondável para baixo then
4      $\bar{x} \leftarrow \bar{x}$  com  $\bar{x}_j$  substituída por  $\lfloor \bar{x}_j \rfloor$ ;
5   else
6     if  $x_j$  é trivialmente arredondável para cima then
7        $\bar{x} \leftarrow \bar{x}$  com  $\bar{x}_j$  substituída por  $\lceil \bar{x}_j \rceil$ ;
8     else
9       parar!;
```

Em (Miller and Franz 1996), os autores propõem uma heurística de arredondamento binário (BRH, do inglês *Binary Rounding Heuristic*) para o problema de alocação de tarefas de duração variável, considerando vários períodos. No problema apresentado, uma tarefa pode requerer mais de um período para ser completada, bem como vários empregados podem ser alocados a uma tarefa ao mesmo tempo. Desta forma, o problema pode ser modelado utilizando variáveis binárias. Se o empregado x está alocado à tarefa y no período z , teremos a variável correspondente ativa. O método proposto no artigo arredonda as variáveis fracionárias para 1, não sendo utilizados arredondamentos para 0.

Dois tipos de inviabilidade podem ocorrer quando se utiliza o arredondamento de variáveis binárias. Primeiro, se muitas variáveis estão ativas no valor 1, o lado direito das restrições pode ser excedido e segundo, uma inviabilidade indireta pode ocorrer quando algumas variáveis são arredondados para o valor 1 e as demais variáveis são incapazes

de satisfazer as restrições do problema. As variáveis a arredondar são cuidadosamente escolhidas de modo que a inviabilidade direta, como descrito acima, seja completamente evitada. Para minimizar a inviabilidade indireta, a heurística é iterativa, arredondando apenas um número limitado de variáveis ao mesmo tempo. Para cada tarefa é calculada uma medida que visa a priorizar quais variáveis devem ser arredondadas primeiro. Assim, durante o processo é provável que soluções inteiras ocorram naturalmente.

Os testes foram realizados utilizando 72 instâncias do problema tratado, comparando-se a heurística proposta com um software de programação linear. O método BRH apresentou bons resultados quando comparado com os valores das soluções resultantes da aplicação do resolvidor de programação inteira SAS diretamente ao problema.

No trabalho (Wallace 2010) é proposta a heurística *ZI Round*, que tenta arredondar cada variável fracionária utilizando as folgas para manter a viabilidade. O objetivo da heurística é procurar as variáveis inteiras que possam ser arredondadas até que a inviabilidade se torne zero em um ponto em que a solução foi encontrada. Para isto calcula para cada variável o quanto está pode ser movida dentro de seu limite, mantendo a viabilidade primal, isto é mantendo todas as folgas não-negativas. Nos experimentos realizados, a heurística foi comparada ao método *Simple Rounding*, sendo utilizadas 59 instâncias da biblioteca MIPLIB 2003. Os resultados obtidos foram satisfatórios, sendo que os dois métodos se comportaram de forma parecida, porém *ZI Round* encontrou um número maior de melhores soluções.

Em (Melo, Nickel and da Gama 2009) é proposto um método de arredondamento para resolver um problema de rearranjo de rede multiperíodo que aparece no contexto do planejamento de uma cadeia de suprimentos. O problema surge quando, por exemplo, oportunidades de expansão aparecem, a demanda em determinados pontos aumenta, ocasionando a necessidade de realocação para áreas mais favoráveis ou mesmo redução de custos. As facilidades, ou centros de distribuição, são modeladas como variáveis binárias, definindo se estão ativas ou não. O algoritmo possui duas fases. Na primeira, uma estratégia de arredondamento é aplicada para determinar as localidades ativas, e neste processo são utilizados quatro critérios de arredondamento. Em seguida, é feita uma busca local para reparar inviabilidades ou mesmo melhorar a solução encontrada. Os testes mostraram que soluções viáveis de boa qualidade foram encontradas para todos os casos, algumas na primeira fase, outras na segunda.

(Ge, He, Ye and Zhang 2011) desenvolveram um método de arredondamento aleatório para uma série de problemas de otimização com restrições de atribuição inteiras. O

núcleo do método é um arredondamento simples, intuitivo, eficiente e geométrico, que arredonda simultaneamente pontos múltiplos em um simplex multidimensional, buscando os vértices. Usando esse método foram obtidos novos resultados para o problema de localização de hub (envolvendo a localização e determinação de tráfego entre os pontos de demanda), determinação do vencedor (em que consiste em alocar itens aos jogadores, de modo a maximizar o valor total) e problemas de rotulagem consistentes (minimizar o número de rótulos por objeto ou maximizar o número de conjuntos consistentemente rotulados). Em geral, o método proposto ofereceu uma alternativa simples e eficaz para os problemas estudados.

2.2 Relaxation Induced Neighborhood Search

A MIP heurística *Relaxation Induced Neighborhood Search* (RINS), é definida no trabalho de (Danna, Rothberg and Le Pape 2003). A ideia do método parte do pressuposto de que ao explorar uma árvore *branch-and-cut*, duas soluções (incumbente e relaxação contínua) estão à nossa disposição. A solução incumbente é viável no que diz respeito às restrições de integralidade, mas ainda não é a solução ideal, haja visto que a solução ótima ainda não tenha sido encontrada. Por outro lado, a solução da relaxação contínua no nó atual, na maioria das vezes não é inteira, mas o seu valor objetivo é sempre melhor do que o valor da solução incumbente. Sendo assim, cada uma delas alcança um dos objetivos e falha no outro: integralidade e valor da função objetivo.

Podemos notar que nestas soluções algumas variáveis claramente possuem valores diferentes, mas também é importante notar que em muitas delas o valor é o mesmo. O método RINS baseia-se na intuição de que as variáveis que possuem o mesmo valor em ambas as soluções formam uma solução parcial do problema. Esta solução parcial pode ser estendida em uma solução completa que atenda a integralidade e melhor valor objetivo, pela resolução das demais variáveis. Assim as variáveis que diferem nas duas soluções serão as que merecem mais atenção. Em cada nó da árvore *branch-and-cut*, os seguintes passos são realizados:

1. fixar as variáveis com o mesmo valor da solução incumbente e a solução relaxada;
2. definir um corte com base no valor da função objetivo na solução incumbente corrente;
3. resolver o sub-MIP com as variáveis restantes.

Podemos perceber que a formulação do problema vai melhorando durante o processo de *branch-and-cut*, com a adição de planos de corte e a descoberta de novos limites globais para as variáveis. O sub-MIP criado se aproveita destas informações adicionais, entretanto os limites acrescentados devido a ramificação não são respeitados para que a pesquisa não fique muito restrita. O método possui uma boa diversificação de vizinhança naturalmente, já que a relaxação muda em cada nó. A relaxação em nós consecutivos é tipicamente muito semelhante, por isso que é preferível aplicá-lo apenas a cada F nós, para algum valor $F \geq 1$. Ainda assim devemos mencionar que é possível, embora improvável, que, mesmo com uma grande variedade de valores para F , vários nós MIP podem produzir os mesmos sub-MIPs, ou da mesma forma que um sub-MIP pode ser idêntico a uma sub-árvore da árvore MIP global.

O sub-MIP a ser resolvido pode ser potencialmente grande e de difícil resolução, sendo assim sua exploração muitas vezes é truncada. Isto é feito pela definição de um número limite de nós a serem explorados. As soluções encontradas para o sub-MIP são sempre soluções válidas para o problema completo. Assim, quando a exploração do sub-MIP termina (se este é inviável ou a otimalidade do sub-problema foi provado), a solução incumbente do MIP é atualizada para a melhor solução inteira encontrada no sub-MIP (se houver), e a resolução do MIP completo é retomada. A robustez do método RINS se deve ao fato deste explorar ao mesmo tempo a vizinhança da solução incumbente e da solução relaxada; além disso, como as soluções possuem papéis simétricos, se uma delas é de má qualidade, a outra irá automaticamente ajudar a definir uma vizinhança mais promissora.

O método RINS é muitas vezes relacionado com o método *Path Relinking* (Glover 1997), porque, de certa forma, ele liga duas soluções encontradas em partes diferentes da árvore *branch-and-cut* pela exploração do espaço de soluções entre elas. Entretanto, se diferencia na medida em que não explora um ou alguns caminhos entre as soluções, mas sim, o sub-espaço de soluções entre elas. Também é mais importante, o método RINS combina soluções relacionadas, porém diferentes (inteira e fracionária).

Os testes foram realizados em 37 instâncias nas quais encontrar uma boa solução viável é difícil. O tempo foi definido de forma que não fosse muito pequeno, neste caso o método não tem oportunidade de terminar, e nem muito grande, pois métodos robustos devem encontrar uma solução o mais cedo possível. Diante disso, foram utilizados diferentes limites de tempo para cada tipo de instância. O método se mostrou o mais efetivo, encontrando boas soluções mais cedo quando comparado aos demais testados (CPLEX, *Local Branching* e *Guided Dives*). Além disso, ele segue a mesma tendência

quando o tempo disponível aumenta.

2.3 Local Branching

Dada uma heurística MIP, analisar o seu comportamento é muito importante: uma estratégia mais agressiva que melhora a solução incumbente nas fases iniciais da computação é preferível a uma estratégia projetada para encontrar boas soluções apenas nos passos finais (que para problemas difíceis não alcança a solução dentro do limite de tempo). Muitos resolvedores MIP permitem ao usuário ter um certo controle sobre seu comportamento heurístico e a ordem de visitaç o da  rvore de *branching*. (Fischetti and Lodi 2003, Fischetti, Polo and Scantamburlo 2004) definem uma estrat gia de explora o da  rvore de *branching*, semelhante a metaheur sticas de busca local, por m as vizinhan as s o obtidas atrav s dos chamados cortes *Local Branching* (LB).

Esta estrat gia atua em dois n veis: Em alto n vel, definindo a vizinhan a de uma solu o, enquanto que em baixo n vel explora as ramifica es em si. Sendo assim, o m todo favorece atualiza es na solu o incumbente em seus passos iniciais, bem como, a produ o de solu es melhoradas na fase final de explora o.

Muitas heur sticas comumente utilizam m todos de fixa o forte de vari veis, que se mostram bastante eficazes; entretanto, um aspecto cr tico est  relacionado com a escolha das vari veis a serem fixadas em cada etapa. Se as escolhas s o ruins e, por conseguinte, afetam outras vari veis que est o relacionadas, boas solu es dificilmente s o encontradas. Resolvendo este tipo de situa o, pode-se utilizar a fixa o leve de vari veis, bastando adicionar ao modelo MIP uma restri o linear suave que evite fixa es r gidas, definindo uma vizinhan a adequada ao problema. Isso pode ser feito pela introdu o de folgas nos valores de fixa o, sendo este mecanismo de fixa o leve utilizado no m todo *Local Branching*.

O m todo *Local Branching* explora para uma solu o, a vizinhan a k -OPT como sendo o conjunto de solu es que satisfazem a uma restri o adicional de *branching* ($H(x, x_*) \leq k$). A restri o define que a dist ncia entre as duas solu es seja no m ximo k , par metro conhecido como raio da vizinhan a.   importante que o valor de k permita sub-problemas que possam ser resolvidos rapidamente mas tamb m contenha diversidade para que boas solu es sejam encontradas. Para melhorar o desempenho do LB s o adicionados par metros relacionados ao tempo e n mero de n s limite explorados na

árvore LB.

Os testes computacionais utilizaram instâncias de difícil resolução encontradas na literatura. Na maioria dos casos testados o resolvidor MIP encontrou soluções significativamente melhores quando guiado pelo LB, apesar de não ser possível provar a otimalidade em tempo razoável.

2.4 Feasibility Pump

A MIP heurística *Feasibility Pump* (FP) foi proposta por (Fischetti, Glover and Lodi 2005). Nesse artigo considera-se o problema de encontrar uma solução viável (se houver) para um problema MIP genérico. No poliedro que representa as soluções de um problema denotamos que um ponto é inteiro se todas as suas componentes são inteiras. Dado um ponto fracionário, um ponto solução mais próximo pode ser encontrado resolvendo o problema linear da distância mínima entre as duas soluções. Também pode ser obtido facilmente através de arredondamentos, atualizando iterativamente de modo a reduzir a distância.

O método FP busca a cada iteração reduzir a distância entre as soluções, de forma que, quando a distância é zero teremos uma solução viável. O método caminha utilizando duas soluções: uma satisfaz as restrições e a outra satisfaz aos requisitos de integralidade.

O pseudocódigo do método FP é apresentado no Algoritmo 2.2.

Algoritmo 2.2: Feasibility Pump

```

1  inicializar  $nIT \leftarrow 0$  ;
2   $x^* \leftarrow \operatorname{argmin}\{c^T x : Ax \geq b\}$  ;
3  if  $x^*$  é inteiro then
4    return  $x^*$ ;
5  tome  $\tilde{x} \leftarrow [x^*]$  (= arredondamento de  $x^*$ );
6  while  $tempo \leq TL$  do
7    tome  $nIT \leftarrow nIT + 1$  e calcule  $x^* \leftarrow \operatorname{argmin} \Delta(x, \tilde{x}) : Ax \geq b$ ;
8    if  $x^*$  é inteiro then
9      return  $x^*$  ;
10   if  $x^*$  é inteiro then
11      $\tilde{x} \leftarrow [x^*]$  ;
12   else
13     inverter  $TT = \operatorname{randômico}(T/2, 3T/2)$  entre  $(j \in \Gamma)$  com o maior  $|x_j^* - \tilde{x}_j|$ ;

```

Inicialmente, no passo 2 do Algoritmo 2.2, obtemos o custo mínimo da solução relaxada. Em seguida são feitos os arredondamentos.

A cada novo ciclo, no passo 7 a distância entre os pontos é redefinida. São executados diversos ciclos até que a solução inteira seja encontrada ou o tempo limite seja alcançado. Há também um método de perturbação, caso o algoritmo entre em *loop*.

Os resultados encontrados pelos autores mostram a efetividade do uso da função de distância para encontrar soluções de qualidade em tempo viável. Variações no método também foram testadas de forma a reduzir o tempo computacional e/ou melhorar a qualidade da solução.

Para (Achterberg and Berthold 2007), encontrar uma solução de alta qualidade de um MIP significa encontrar um ponto x que satisfaça a três requisitos: x pertence ao conjunto de soluções, as variáveis são inteiras e o custo é o menor possível. O FP gera sequências de pontos que satisfazem ao primeiro e ao segundo critérios, até que se encontre um ponto que satisfaça a ambos. Apesar do cálculo do ponto de partida ser escolhido como o ótimo da relaxação LP, a terceira condição é ignorada. Nesta abordagem, conhecida como *Objective Feasibility Pump*, a função objetivo original do MIP não é descartada de imediato, sua influência no método é reduzida gradualmente,

enquanto o peso da função de distância aumenta. Assim, a modificação consiste em substituir o uso apenas da função de distância, por sua combinação com a função objetivo do problema. Pelos resultados pode-se verificar uma melhora no método FP, sendo que a qualidade das soluções resultantes é significativamente melhorada com apenas um ligeiro aumento no tempo de execução.

(Baena and Castro 2011) discutem o uso de um centro analítico aplicado ao FP. O método criado foi denominado *Analytic Center Feasibility Pump* (AC-FP). O artigo sugere uma extensão do método onde todos os pontos de um possível segmento de soluções viáveis são candidatos a serem arredondados. No método FP padrão, o ponto a ser escolhido e arredondado é sempre o ponto final do segmento, enquanto utilizando o centro analítico um ponto no interior do polítopo deverá ser o melhor candidato.

O centro analítico fornece o ponto que maximiza a distância para os hiperplanos que delimitam o polítopo e, portanto, espera-se que seja bem centrado no interior deste. Um mesmo polítopo pode ser definido por diferentes desigualdades lineares e, neste caso, os centros serão diferentes. O arredondamento de um ponto do segmento mais próximo ao centro analítico pode aumentar as chances de obter uma solução inteira, e em alguns casos, reduzir o número de iterações do FP.

Diferentemente de métodos semelhantes encontrados na literatura, neste caso o cálculo do centro analítico acontece apenas na primeira iteração, tornando o método mais simples. O método AC-FP supera em alguns casos o FP (tanto em termos de tempo quanto em qualidade da solução); no entanto, para os problemas binários AC-FP não mostrou ser competitivo.

No trabalho (Bonami, Cornuéjols, Lodi and Margot 2009) uma heurística para encontrar soluções viáveis para o problema de programação não linear mista inteira (MINLP) é definida. O princípio básico de funcionamento do *Feasibility Pump* é gerar uma sequência de pontos que satisfazem a relaxação contínua. Associado a esta sequência de pontos inviáveis estão pontos que estão na solução inteira mas não necessariamente satisfazem as restrições do problema. Estes pontos são arredondamentos e gerados resolvendo um problema linear cujo objetivo é minimizar a distância entre as soluções. A cada iteração a distância entre as sequências vai diminuindo. Se ocorre ciclagem, são utilizados mecanismos que permitem o reinício. Para os problemas não lineares, são também construídas duas sequências, que porém satisfazem requisitos diferentes.

FB para MINLP é uma heurística que em regiões convexas se comporta como um algoritmo *Outer Approximation*. Neste caso não ocorre ciclagem e o método termina em um

número finito de passos, quando todas as variáveis estão delimitadas. Assim, encontra-se uma solução viável quando existe ou prova-se que não existe solução. O método FB aplicado a MINLP foi implementado em COIN-Ipopt3 para problemas não lineares e Cplex para problemas lineares inteiros. Os resultados mostraram que a combinação OA+FB se mostrou eficiente encontrando a solução ótima na maioria dos casos.

Em (Fischetti and Salvagnin 2009) é apresentada uma evolução do método FP. Os autores acreditam que a principal desvantagem do esquema básico do FP é a sua tendência a estagnar, caso em que uma perturbação aleatória (ou mesmo uma reinicialização) é realizada na tentativa de escapar do ótimo local. Então na tentativa de melhorar a taxa de sucesso do método, observa-se que o FP pode ser visto como uma maneira inteligente de arredondar uma sequência de soluções fracionárias do problema, até que uma solução viável é finalmente encontrada. Desta forma, é natural tentar substituir a operação de arredondamento original (que é rápida e simples, mas de alguma forma cega) por uma heurística de arredondamento mais inteligente.

O procedimento utilizado nesta evolução do método é baseado em busca em profundidade com arredondamento e propagação de restrições. Propagação de restrições é basicamente uma forma de inferência que consiste em proibir explicitamente valores, ou combinações de valores, para algumas variáveis do problema. Esta integração é baseada na seguinte observação: Arredondar uma variável significa temporariamente corrigi-la a um determinado valor, então podemos em princípio, propagar esta fixação temporária antes de arredondar as demais variáveis. Também no FP a aplicação da propagação é feita até o fim, porque temos de escolher um valor para todas as variáveis inteiras.

Os resultados computacionais, tanto para instâncias binárias quanto inteiras, mostraram que este método encontra um maior número de soluções viáveis do que seu antecessor, e em menor tempo computacional, sendo a principal característica a redução do número de iterações para atingir a convergência.

2.5 Proximity Search

A heurística *Proximity Search* (PS) foi definida no trabalho de (Fischetti and Monaci 2012), e discute o papel da função objetivo na otimização. A importância da função objetivo, dentre outras, é guiar a busca de uma solução viável na árvore de *branch-and-bound*. Assim, nesse trabalho, os autores investigam os efeitos em se trocar esta função

de custo por uma função de proximidade.

Foram utilizados problemas MIP binários, com funções de restrição convexas. O método basicamente segue os seguintes passos/etapas:

1. Encontrar uma solução inicial viável através do *branch-and-bound*;
2. Adicionar, em seguida, um corte baseado no valor da função objetivo atual, e definir também um percentual de tolerância;
3. Calcular a função de proximidade; neste caso, pela distância de Hamming;
4. Calcular a nova solução incumbente, e o reiniciar o método a partir desta, até que o critério de parada seja atendido.

A ideia de trabalhar com funções de proximidade não é nova. A heurística *Local Branching* utiliza de certa forma este modelo, onde através da adição de uma restrição é definido o tamanho da vizinhança próxima que será pesquisada. No método PS comparado ao LB, a proximidade passa a ser uma função e não mais uma restrição. No LB procura-se apenas perto da raiz, devido a limitação de vizinhança, enquanto no método PS pode-se pesquisar em todo o espaço de busca.

São propostas também duas variações para este método:

1. PS sem *recentering*: Neste caso, a função de proximidade é calculada apenas uma vez, sendo este valor sempre o mesmo;
2. PS com *recentering*: Sempre que uma nova solução incumbente é encontrada, os valores da função de busca de proximidade são atualizados.

Nos testes foram utilizados três conjuntos de teste: *Set Covering* (puramente 0-1), *Network Design* (0-1 e contínua), *Machine Learning* (não-linear). Como métrica de avaliação é utilizada uma medida entre o esforço para obter uma solução e o tempo. Um ponto positivo da heurística PS é que para muitos casos a função de proximidade é calculada de forma muito mais rápida do que a função objetivo original; assim, a solução inteira é atualizada mais vezes num mesmo tempo. Um aspecto negativo é a necessidade da adição do corte, que pode ser muito restritivo para alguns casos. Os resultados foram bem promissores, evidenciando que esta troca pode ser eficaz.

2.6 Evolutionary Algorithm

Os algoritmos evolucionários adotam analogia com a seleção natural, explorando conceitos como população, combinação, mutação e seleção para um espaço diverso de soluções possíveis para os problemas de otimização, enquanto, ao mesmo tempo, retêm propriedades desejáveis das soluções conhecidas. (Rothberg n.d.) descreve uma abordagem evolucionária aplicada a problemas MIP. Esses métodos tipicamente mantêm uma população de soluções, e em cada fase do algoritmo uma nova geração de soluções é criada a partir da geração anterior. A nova população é formada a partir de três operações:

1. Seleção: Soluções candidatas são escolhidas, baseado em uma métrica de desempenho, para formar a próxima geração ou para serem combinadas;
2. Combinação: Pares de solução são combinadas, gerando novas soluções;
3. Mutação: Mudanças aleatórias são introduzidas em uma solução, de forma a gerar diversidade no método.

O método descrito por (Rothberg n.d.) implementa o conceito de população, mantendo um *pool* de tamanho fixo contendo as melhores P soluções. A operação de combinação foi definida com características semelhantes ao RINS (Danna, Rothberg and Le Pape 2003), onde neste caso são escolhidas duas soluções e fixadas as variáveis que possuem o mesmo valor. O sub-problema encontrado é então resolvido e a melhor solução é armazenada no *pool*. A operação de seleção das combinações ocorre da seguinte forma: dado o *pool* de soluções, estas são classificadas em ordem de valor objetivo (com a idade da solução, como chave secundária), em seguida escolhe-se uma solução aleatória do conjunto, como primeiro pai, e depois escolhe-se o segundo pai aleatoriamente dentre aqueles que vêm antes do primeiro na lista ordenada. A operação de mutação consiste em selecionar uma solução e fixar um percentual de variáveis, então o sub-problema gerado é resolvido, o percentual de fixação é atualizado e a solução é armazenada no *pool*.

Os testes realizados comparam o desempenho do método proposto, do CPLEX, e da aplicação das heurísticas (RINS, *Local Branching* e *Guided Dives*). Os resultados mostram que o método obteve resultados satisfatórios.

2.7 Relaxation Enforced Neighborhood Search

A abordagem *Local Search* generaliza a ideia de k -OPT, ou seja, define uma vizinhança de algum ponto de referência e determina um ponto ideal nesta vizinhança de acordo com alguma função (por exemplo, a função objetivo do MIP ou alguma medida de viabilidade). Este ponto é então utilizado como referência para a próxima iteração. Desta forma a heurística *Local Search* utiliza de vizinhanças relativamente pequenas que podem ser rapidamente exploradas.

Uma nova técnica vem sendo utilizada recentemente, conhecida como *Large Neighborhood Search* (LNS), que é uma variante do método *Local Search*. O método LNS incorpora a complexidade das problemas MIPs definindo uma vizinhança relativamente grande de algum ponto, normalmente a solução incumbente, e em apenas uma iteração a vizinhança é completamente ou parcialmente pesquisada. Diversas heurísticas baseadas em LNS têm sido propostas. Elas têm em comum o fato de usar a solução incumbente como ponto de partida e em seguida definir a vizinhança como um sub-MIP do problema original através da fixação de variáveis e adição de restrições.

A heurística *Relaxation Enforced Neighborhood Search* (RENS) (Berthold 2008) propõe utilizar apenas o ótimo da relaxação, não sendo necessária uma solução inicial inteira. RENS é uma heurística LNS que investiga todos os arredondamentos possíveis da solução do relaxamento LP. Na solução relaxada, muitas vezes um conjunto de variáveis já possui valor inteiro, então a ideia é fixar estas variáveis e realizar uma busca LNS sobre as demais variáveis com os valores arredondados para o inteiro mais próximo.

O método RENS pode ainda ser utilizado para analisar outras heurísticas de arredondamento. Se o subproblema criado é provado como inviável, não existe heurística de arredondamento capaz de gerar uma solução viável. Entretanto, se o subproblema está completamente resolvido pelo RENS, essas soluções ótimas são os melhores arredondamentos que qualquer heurística de arredondamento puro pode gerar. Os resultados apresentados revelam que a heurística possui um desempenho superior ao do resolvidor puro.

Capítulo 3

pRINS: A heurística BP proposta

Neste capítulo descrevemos a heurística proposta para resolver problemas binários (heurística BP). Na seção 3.1 é apresentada a heurística RINS (Danna, Rothberg and Le Pape 2003), que é a base do método proposto, enquanto na seção 3.2 é revisto o método de busca binária que é utilizado no pRINS. Na seção 3.3 é apresentada a técnica de pré-processamento do resolvidor CBC e, finalmente, na seção 3.4 o algoritmo proposto é explicado em detalhes.

A Figura 3.2 apresenta um esquema geral da heurística BP proposta, denominada pRINS. O método é uma alternativa eficaz à dos métodos clássicos, tais como *branch-and-bound* e *branch-and-cut*. O método parte de duas soluções (inteira e fracionária), em seguida cria um vetor contendo as prioridades de fixação de variáveis considerando estas soluções. Dado um tamanho de problema desejado, procura-se um número de fixações que encontre este tamanho (Algoritmo 3.1), esta variação no número e forma com que as variáveis são fixadas dá ao método uma maior flexibilidade, se comparado à heurística de (Danna, Rothberg and Le Pape 2003). Em seguida, é construído e resolvido o sub-problema binário (Algoritmo 3.2). Se ao resolvê-lo for encontrada uma solução melhor, então são atualizadas as soluções correntes e o tamanho desejado e recalculado o vetor de prioridades. Caso contrário, somente o tamanho buscado é atualizado. O método executa enquanto o tamanho de problema máximo não é atingido ou o tempo se esgote.

Os algoritmos 3.1 e 3.2 serão melhor definidos nas próximas seções.

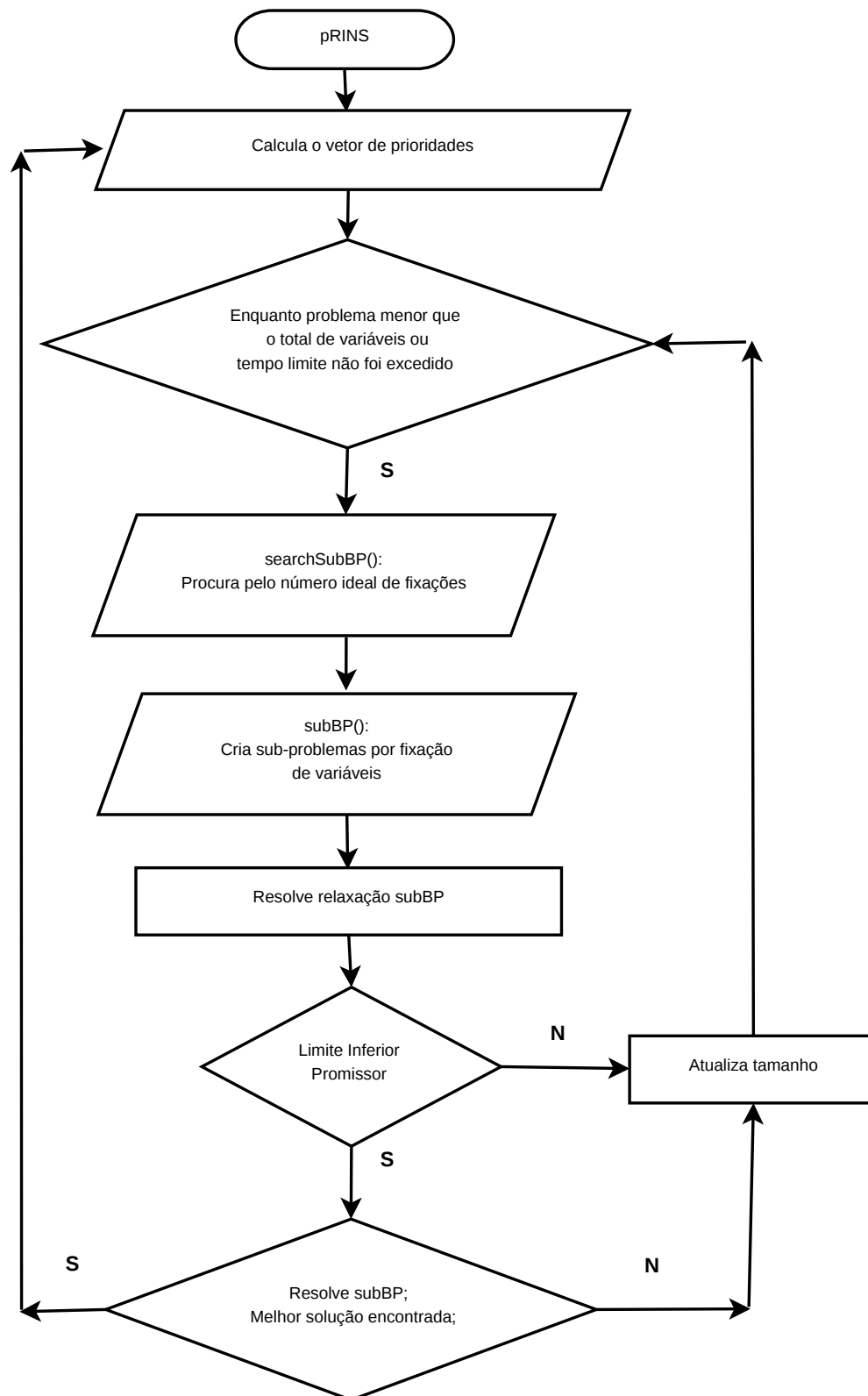


Figura 3.1: Esquema Geral do método pRINS

3.1 RINS

A heurística desenvolvida neste trabalho é baseada no método RINS, apresentado na seção 2.2 e revisto na seção corrente. Nesse método, em cada nó da árvore de *branch-and-cut* as seguintes operações são realizadas:

1. fixar as variáveis com o mesmo valor na solução incumbente e na solução relaxada;
2. definir um corte com base no valor da função objetivo na solução incumbente corrente;
3. resolver o sub-MIP com as variáveis restantes.

Assim, o método RINS fixa todas as variáveis que são iguais entre as duas soluções. Esta estratégia pode gerar ora sub-problemas com muitas fixações (assim o espaço de soluções fica pequeno), ou sub-problemas com pouca fixação (com espaço de soluções extenso). No exemplo da Figura 3.2, serão fixadas as variáveis marcadas em azul e em seguida o problema deve ser resolvido com as variáveis restantes.

3.2 Busca Binária

A pesquisa ou busca binária (*binary search algorithm* ou *binary chop*) é um algoritmo de busca em vetores que segue o paradigma de divisão e conquista. Ela parte do pressuposto de que o vetor está ordenado e realiza sucessivas divisões do espaço de busca comparando o elemento buscado (chave) com o elemento no meio do vetor. Se o elemento do meio do vetor for a chave, a busca termina com sucesso. Caso contrário, se o elemento do meio vier antes do elemento buscado, então a busca continua na metade posterior do vetor. E finalmente, se o elemento do meio vier depois da chave, a busca continua na metade anterior do vetor.

Neste trabalho, a pesquisa binária visa encontrar o número de fixações necessário para atingir o tamanho de problema inicialmente definido, onde o universo de pesquisa varia de zero ao número total de variáveis. Esta idéia é utilizada no Algoritmo 3.1, apresentado a seguir.

O método *searchSubBP*, definido no Algoritmo (?) é usado para determinar o número ideal de fixações. Dado o tamanho desejado do problema ($nDes$), que no início da

execução é um tamanho de problema suficientemente pequeno para que possa ser resolvido rapidamente, o número total de variáveis ($\text{size}(\bar{x})$), número máximo de tentativas (maxTries), o vetor de prioridade das fixações (v), o vetor de variáveis inteiras ($\bar{x}$), o problema binário $\mathcal{P}$, então os sub-problemas binários são criados e o número de fixações necessário é encontrado.

Para alcançar o tamanho desejado, o número de fixações vai sendo testado através de uma busca binária até que o número máximo de tentativas se esgote, a busca binária termine ou o tamanho seja encontrado.

Inicialmente o limite superior de fixações ($\bar{n}$) é o número total de variáveis do problema. Este limite é modificado ao final de cada busca binária, sendo atualizado para o número de fixações utilizado no último sub-problema resolvido. O método invoca uma função para criar os problemas (definido em 3.2), fixando as variáveis e checando a viabilidade do sub-problema resultante. Depois de construído o problema binário, a função *vars* retorna o número de variáveis livres neste sub-problema. A busca termina quando o número de variáveis livres ($\text{vars}(\mathcal{P}')$) atinge o tamanho desejado ($nDes$), dado um percentual de tolerância (d). De outra forma, a busca termina se o número máximo de tentativas (maxTries) é alcançado, ou o limite superior ($\bar{n}$) é menor que o limite inferior

$(\underline{n})$.

Algoritmo 3.1: searchSubBP

Input: $\bar{x}, v, maxTries, nDes, d, \mathcal{P}, F$

Output: N

```

1 //  $\bar{x}$ : integer solution;
2 //  $v$ : set of priorities;
3 //  $maxTries$ : number of attempts ;
4 //  $nDes$ : desired size ;
5 //  $d$ : percentage tolerance ;
6 //  $\mathcal{P}$ : binary problem;
7 //  $F$ : set of indexes of variables to be fixed;
8  $\underline{n} = 1$ ;
9  $\bar{n} = \text{size}(\bar{x})$ ;
10  $tries = 0$ ;
11 while  $++tries \leq maxTries$  or  $\bar{n} > \underline{n}$  do
12    $n^* = \lfloor (\underline{n} + \bar{n})/2 \rfloor$ ;
13    $F \leftarrow$  a set of elements built by tournament, greedy or shift methods ;
14    $\mathcal{P}' \leftarrow \text{subBP}(\bar{x}, F, \mathcal{P})$ ;
15   if  $|\text{vars}(\mathcal{P}') - nDes| \leq nDes * d$  then
16     return  $\mathcal{P}'$ ;
17   if  $\mathcal{P}' = \text{NULL}$  or  $\text{vars}(\mathcal{P}') < n$  or  $f(\text{relax}(\mathcal{P}')) \geq f(\mathcal{P})$  then
18      $\bar{n} = n^* - 1$ ;
19   else
20      $\underline{n} = n^* + 1$ ;

```

A Figura 3.3 traz um esquema do método *searchSubBP*, que implementa a busca binária.

3.3 Pré-processamento

O objetivo desta técnica é reduzir o tamanho do problema e melhorar a formulação. Embora possa retardar o início da pesquisa, o modelo resultante pode ser muito mais fácil de resolver (Santos 2011). O COIN-OR CBC decide de que forma o pré-processamento será realizado.

Para o método proposto a técnica indica se o problema pré-processado é viável ou inviável, retornando o número de variáveis livres quando este for viável e zero caso contrário, conforme esquematizado na Figura 3.4.

O pré-processamento é utilizado nos métodos que criam os sub-problemas binários. A criação de problemas é apresentada no Algoritmo 3.2.

Algoritmo 3.2: subBP

Input: $\bar{x}, F, \mathcal{P}$

Output: $\mathcal{P}''$

```

1 //  $\bar{x}$ : integer solution;
2 //  $F$ : set of indexes of variables to be fixed;
3 //  $\mathcal{P}$ : binary program;
4  $\mathcal{P}' \leftarrow \mathcal{P}: x_i = \bar{x}_i \ \forall i \in F$ ;
5  $\mathcal{P}'' \leftarrow \text{preProcess}(\mathcal{P}')$ ;
6 if unfeasible( $\mathcal{P}''$ ) then
7   | return NULL;
8 else
9   | return  $\mathcal{P}''$ ;

```

O método subBP, descrito pelo Algoritmo 3.2, constrói sub-problemas binários, considerando as variáveis a serem fixadas que estão definidas no vetor de índices (F). Os limites superiores e inferiores das variáveis são limitados pelos valores contidos na solução inteira $\bar{x}$. O sub-problema criado $\mathcal{P}'$ é pré-processado, gerando um novo ($\mathcal{P}''$). Se o problema binário ($\mathcal{P}''$) é viável, o método retorna o tamanho deste. Se este é inviável, retorna 0 como tamanho. O vetor de índices F é definido por três maneiras: 1) sequencialmente por ordem de prioridade, 2) escolhidas por métodos de torneio (3.3) e 3) escolhidas por um método de *shift* (3.4). O esquema do método que cria os sub-bps pode ser observado na Figura 3.5.

No Algoritmo 3.3, dado um número de fixações N , o vetor de prioridades v e a variável booleana r , constrói-se e retorna-se um vetor de índices F . Esse vetor contém as posições das variáveis que serão fixadas. O booleano r é utilizado para definir se o vetor índice criado pelo método será guloso ou determinado através de torneio. O método consiste em construir um conjunto de variáveis a serem fixadas dando preferência de inclusão as variáveis que estão com maior prioridade. São selecionados aleatoriamente alguns elementos, com igual chance de serem escolhidos, e o melhor deles é alocado no vetor

de índices. No presente trabalho foram testados quatro variações de torneio a saber: binário, ternário, quaternário e quinquenário.

Algoritmo 3.3: randomBiasedConstruction

Input: v, N, r

Output: F

```

1 //  $v$ : set of priorities;
2 //  $N$ : number of fixations;
3 //  $r$ : boolean ;
4 for  $i \leftarrow 1$  to  $N$  do
5   if  $r$  then
6      $a \leftarrow$  element randomly selected in  $v$  ;
7      $b \leftarrow$  element randomly selected in  $v$  ;
8      $F \leftarrow F \cup \text{best}(a,b)$ ;
9   else
10     $F \leftarrow F \cup v(i)$ 
11 return  $F$ ;

```

O outro método para a definição dos índices a fixar é definido pelo Algoritmo 3.4, onde são dados um número de fixações N , o vetor de prioridades v , a variável booleana r , o percentual de troca $shiftP$ e a posição i a ser trocada. Também neste caso o booleano

r define se o vetor de índices será criado de forma gulosa ou usando o *shift*.

Algoritmo 3.4: shiftedConstruction

Input: $v, N, i, shiftP, r$

Output: F

```

1 //  $v$ : set of priorities;
2 //  $N$ : number of fixations;
3 //  $i$ : number of iterations;
4 //  $shiftP$ : percent of shift;
5 //  $r$ : boolean ;
6 if  $r$  then
7    $shiftS = \text{size of } v / shiftP$  ;
8    $posMax = i * shiftS$  ;
9    $posMin = (i - 1) * shiftS$  ;
10   $F \leftarrow$  add the  $N$  firsts elements of  $v$ , excluding those between positions
     $posMax$  and  $posMin$  ;
11   $F \leftarrow$  add the  $shiftS$  next elements of  $v$ , after position  $N$  ;
12 else
13    $F \leftarrow$  add the  $N$  firsts elements of  $v$  ;
14 return  $F$ ;

```

No Algoritmo 3.4 é calculado, inicialmente, o tamanho da troca, que está associado ao tamanho do vetor e ao percentual *shiftP*. Em seguida, devemos calcular a parte do vetor que será trocada, definindo as posições *posMax* e *posMin*, sendo que estas posições dependem de qual iteração de troca (i) estamos executando. Por fim, são adicionadas a F as N primeiras posições do vetor de prioridade, excluindo aquelas entre *posMax* e *posMin*. Depois são acrescentadas ao vetor as *shiftS* posições após N . Sendo assim, o método constrói, a cada iteração, de forma determinística, o vetor de índices a fixar. Foram testados os seguintes percentuais de troca: 10%, 25% e 50%.

3.4 Método pRINS Proposto

O método pRINS (*Pre-processing aware RINS*) proposto explora técnicas de pré-processamento existentes para gerar rapidamente sub-problemas com tamanho controlado. Esses sub-problemas são resolvidos de maneira semelhante ao método *Variable Neighborhood Des-*

cent, VND (Hansen and Mladenović 1997).

A seguir, são apresentados três variantes do método pRINS, cada qual diferenciando-se pelo critério de escolha das variáveis a serem fixadas no vetor de prioridades, tal como definido na seção anterior.

O método pRINS1, dado pelo Algoritmo 3.5, estabelece um critério de parada na aplicação desta heurística. Considera-se uma solução inteira ($\bar{x}$), uma solução fracionária ($\underline{x}$), o tamanho inicial de sub-problema desejado ($nDes$), o número máximo de tentativas para construir um sub-problema no tamanho desejado ($maxTries$), um percentual de diferença aceitável entre o tamanho desejado e encontrado (d) e o problema binário original ($\mathcal{P}$). Inicialmente, é criado um vetor ordenado indicando as prioridades de fixação. Este é definido calculando-se a diferença em magnitude entre os valores da solução inteira e fracionária. As variáveis cuja diferença entre as soluções ($\bar{x}$) e ($\underline{x}$) é zero ficam no início do vetor, enquanto as demais ficam ordenadas pela diferença

ascendente dos valores.

Algoritmo 3.5: pRINS1

Input: $\underline{x}, \bar{x}, \mathcal{P}, nDes, maxTries, d$

Output: Sol^*

```

1  $Sol^* \leftarrow \bar{x}$ ;
2  $v$  receives the sort of the variables according to their priority for fixing considering
   the integer and fractional solution  $(\bar{x}, \underline{x})$ ;
3 repeat
4    $N \leftarrow searchSubBP(\bar{x}, v, maxTries, nDes, d, Lim_{\mathcal{P}})$ ;
5    $F \leftarrow N$  firsts elements of vector  $v$  ;
6    $\mathcal{P}' \leftarrow subBP(\bar{x}, F, \mathcal{P})$ ;
7   if  $\mathcal{P}'$  relaxation cost indicates possible improvement then
8      $Sol_{rins} \leftarrow solve \mathcal{P}'$ ;
9     if found better solution then
10       $Sol^* \leftarrow Sol_{rins}$ ;
11      Recalculate the vector of priorities ( $v$ ) considering  $Sol^*, \underline{x}$  ;
12   else
13     Increase the required size  $nDes$ ;
14   else
15     Increase the required size  $nDes$ ;
16 until the required size is smaller than the total number of variables and time
    limit not exceeded ;
17 return  $Sol^*$ ;

```

O método *searchSubBP*, definido pelo Algoritmo 3.1, é usado para determinar o número de fixações (N) necessárias para atingir o tamanho de sub-problema desejado. Nele é criado um novo sub-problema($\mathcal{P}'$), que em seguida é pré-processado, considerando o número de fixações determinado previamente. Se o custo da relaxação (obtido no pré-processamento) não indicar possibilidade de melhora, então o tamanho desejado é atualizado e incrementado. Se existe possibilidade de melhora, considerando o valor da relaxação, o sub-problema é resolvido. Ao resolver, se uma solução melhor é encontrada, a solução inteira é atualizada e as prioridades são recalculadas. De outra maneira, se a solução encontrada é de piora, o tamanho desejado é atualizado e incrementado ($nDes$) em um percentual para considerar um espaço de busca maior.

As figuras 3.6 e 3.7 esquematizam o funcionamento das variantes do método pRINS

proposto. Na Figura 3.6 representa-se a fase 1 do método, onde temos a entrada de dados, a criação do vetor de prioridades, a busca pelo número de fixações ideal e em seguida a criação do sub-problema binário. Na fase 2 do método, dada pela Figura 3.7, é definida a política de incremento no tamanho do problema a ser resolvido.

Uma variação do método pRINS foi desenvolvida e denominada pRINS2, apresentada no Algoritmo 3.6. Nesta versão foi utilizado o método de construção aleatória com torneios, com igual probabilidade de sorteio de cada um dos elementos (Algoritmo 3.3) para definir as variáveis a serem fixadas. Armazenadas as posições sorteadas em um vetor de índices F , em seguida os problemas são criados através do Algoritmo 3.2. Assim, os sub-problemas são criados e resolvidos, sendo os tamanhos de problema atualizados se

não houver melhora.

Algoritmo 3.6: pRINS2

Input: $\underline{x}, \bar{x}, \mathcal{P}, nDes, maxTries, d$

Output: Sol^*

```

1  $Sol^* \leftarrow \bar{x}$ ;
2  $v$  receives the sort of the variables according to their priority for fixing considering
   the integer and fractional solution  $(\bar{x}, \underline{x})$ ;
3 repeat
4    $N \leftarrow \text{searchSubBP}(\bar{x}, v, maxTries, nDes, d, Lim_{\mathcal{P}})$ ;
5    $i \leftarrow 0$ ;
6   repeat
7      $i \leftarrow i + 1$ ;
8     if  $i == 1$  then
9        $F = \text{randomBiasedConstruction}(v, N, false)$  ;
10    else
11       $F = \text{randomBiasedConstruction}(v, N, true)$  ;
12     $\mathcal{P}' \leftarrow \text{subBP}(\bar{x}, F, \mathcal{P})$ ;
13    if  $\mathcal{P}'$  relaxation cost indicates possible improvement then
14       $Sol_{rins} \leftarrow \text{solve } \mathcal{P}'$ ;
15      if found better solution then
16         $Sol^* \leftarrow Sol_{rins}$ ;
17        Recalculate the vector of priorities  $(v)$  considering  $Sol^*, \underline{x}$  ;
18         $i \leftarrow 0$ ;
19
20
21  until the number of iterations or time limit not exceeded ;
22  Increase the required size  $nDes$ ;
23 until the required size is smaller than the total number of variables and time
   limit not exceeded ;
24 return  $Sol^*$ ;

```

Outra versão do método está definida e apresentada no Algoritmo 3.7. O algoritmo pRINS3 sorteia as variáveis que serão fixadas considerando a técnica de *shift* ou trocas (Algoritmo 3.4). Com as variáveis escolhidas e alocadas no vetor de índices F , os sub-problemas binários são criados utilizando o Algoritmo 3.2, sendo em seguida resolvidos,

onde o tamanho de problema desejado é atualizado se não houver melhora.

Algoritmo 3.7: pRINS3

Input: $\underline{x}, \bar{x}, \mathcal{P}, nDes, maxTries, d$

Output: Sol^*

```

1  $Sol^* \leftarrow \bar{x}$ ;
2  $v$  receives the sort of the variables according to their priority for fixing considering
   the integer and fractional solution  $(\bar{x}, \underline{x})$ ;
3 repeat
4    $N \leftarrow \text{searchSubBP}(\bar{x}, v, maxTries, nDes, d, Lim_{\mathcal{P}})$ ;
5    $i \leftarrow 0$ ;
6   repeat
7      $i \leftarrow i + 1$ ;
8     if  $i == 1$  then
9        $F = \text{shiftedConstruction}(v, N, i, shiftP, false)$  ;
10    else
11       $F = \text{shiftedConstruction}(v, N, i, shiftP, true)$  ;
12     $\mathcal{P}' \leftarrow \text{subBP}(\bar{x}, F, mip)$ ;
13    if  $\mathcal{P}'$  relaxation cost indicates possible improvement then
14       $Sol_{rins} \leftarrow \text{solve } mip''$ ;
15      if found better solution then
16         $Sol^* \leftarrow Sol_{rins}$ ;
17        Recalculate the vector of priorities ( $v$ ) considering  $Sol^*, \underline{x}$  ;
18         $i \leftarrow 0$ ;
19
20
21  until the number of iterations or time limit not exceeded ;
22  Increase the required size  $nDes$ ;
23 until the required size is smaller than the total number of variables and time
   limit not exceeded ;
24 return  $Sol^*$ ;

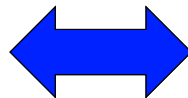
```

Solução Inteira

1
0
0
1
0
0
0
1
1
0

Solução Fracionária

1
0,3
0,5
1
0
0,6
0,3
0,5
1
0



Fixa as variáveis de mesmo valor;
Resolve o sub-problema;

Figura 3.2: Método RINS

Solução Inteira

0	1	0	1	0	0
---	---	---	---	---	---

Solução Fracionária

1	1	0,2	0	0,5	0
---	---	-----	---	-----	---

Tamanho Desejado

100

Vetor de Prioridades

x4	x2	x3	x5	x6	x1
0	0	0,2	0,5	1	1

Número de Tentativas

25

Número Total de Variáveis

930

(SearchSubBP)

Neste caso, a busca binária (0 - 930 variáveis) irá determinar o número ideal de fixações para alcançar o tamanho desejado em 25 tentativas.

Número de Fixações

330

Figura 3.3: Esquema do método searchSubBP

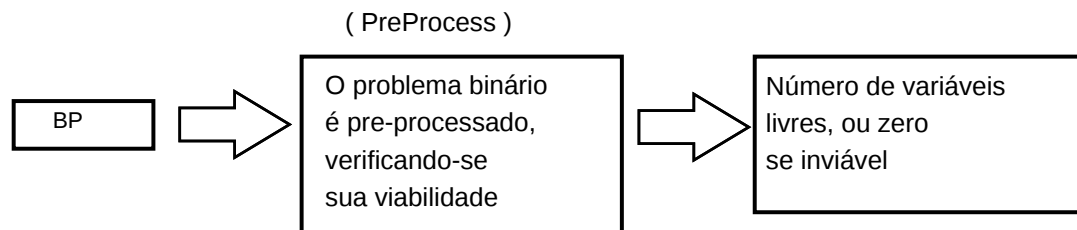


Figura 3.4: Pré-processamento

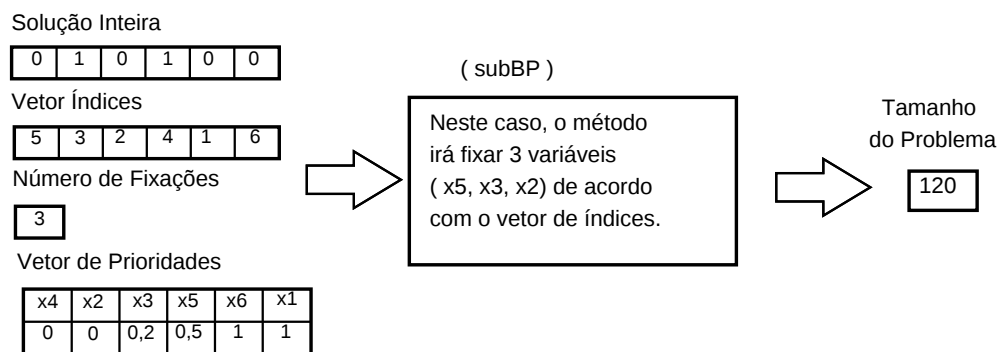


Figura 3.5: Esquema do método SubBP

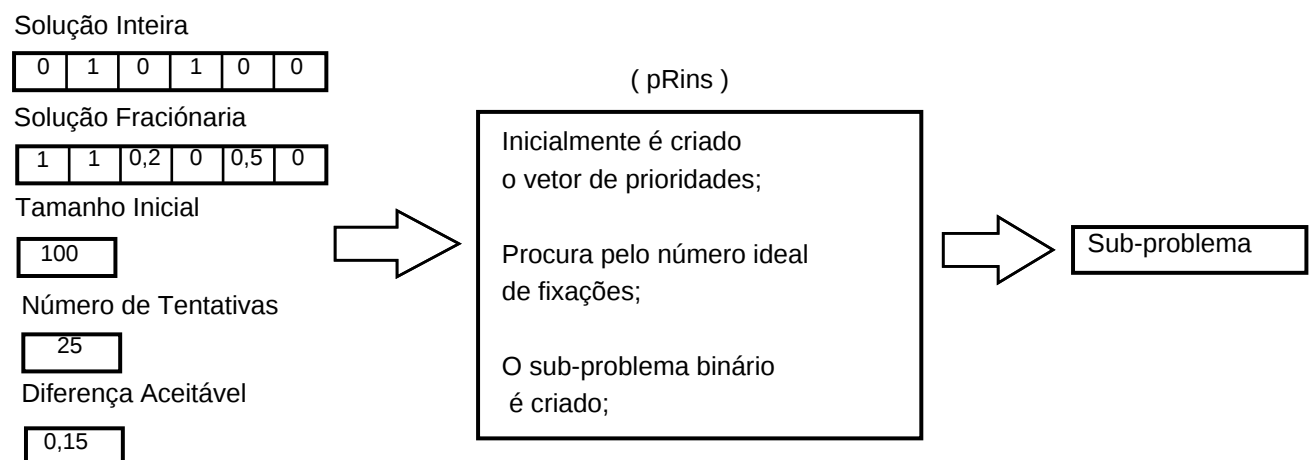


Figura 3.6: Esquema da fase 1 da heurística pRINS

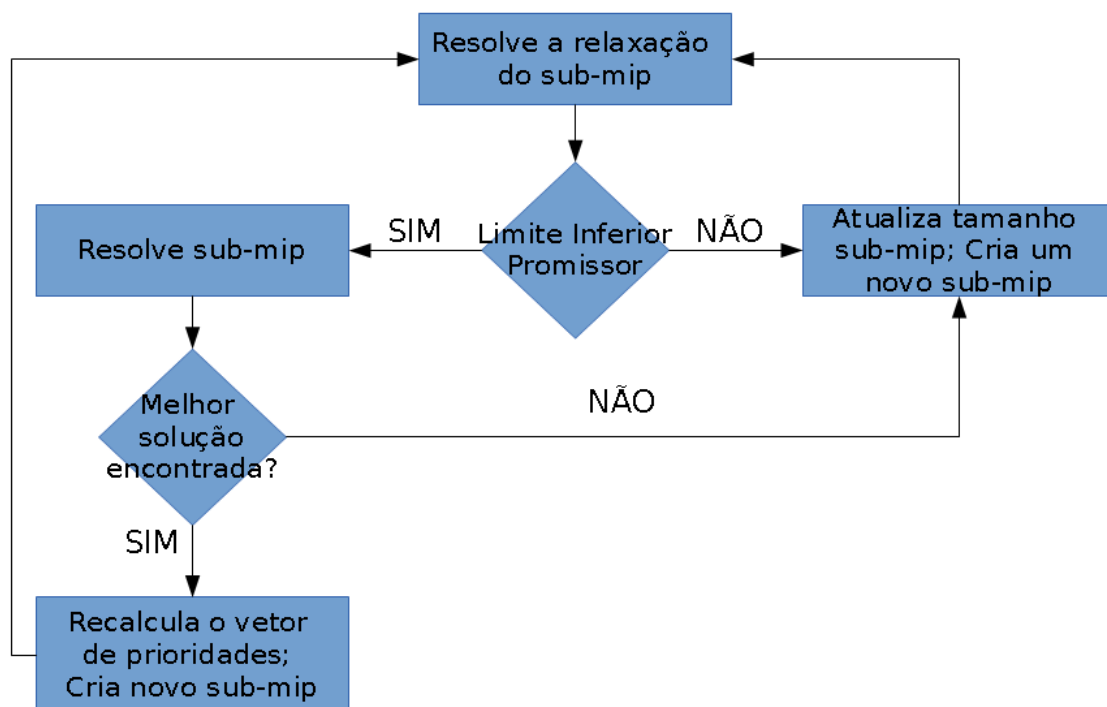


Figura 3.7: Esquema da fase 2 da heurística pRINS

Capítulo 4

Experimentos Computacionais

Neste capítulo são apresentados os experimentos computacionais. Na seção 4.1 o ambiente computacional de desenvolvimento e testes é apresentado, enquanto que na seção 4.2 as instâncias utilizadas são descritas. Finalmente, na seção 4.3 os resultados e experimentos são detalhados.

4.1 Ambiente de Desenvolvimento

As variantes pRINS1, pRINS2 e pRINS3 do método pRINS proposto foram implementadas utilizando o sistema operacional Ubuntu Linux, a linguagem de programação C++, o ambiente de desenvolvimento CodeBlocks, o resolvidor COIN-OR CBC e suas bibliotecas. Os testes foram executados usando a seguinte configuração de hardware: Intel (R) Core (TM) i7 CPU, 1.90GHz, 6 GB RAM.

4.2 Caracterização das Instâncias

Os modelos (instâncias) utilizados neste trabalho são relacionados a problemas binários diversos, a seguir descritos:

- 14 problemas binários da biblioteca MIPLIB 3 <http://miplib.zib.de/> (Koch, Achterberg, Andersen, Bastert, Berthold, Bixby, Danna, Gamrath, Gleixner, Heinz, Lodi, Mittelman, Ralphs, Salvagnin, Steffy and Wolter 2011);

Tabela 4.1: Instâncias MIPLIB 3

instância	restrições	var. binárias	não-zeros	solução ótima
air03	124	10.757	91.028	340.160
air04	823	8.904	72.965	56.137
air05	426	7195	52121	26.374
cap6000	2.176	6.000	48249	-2.451.377
harp2	112	2.993	5.840	-73.899.798
l152lav	97	1.989	9.922	4.750
lseu	28	89	309	1.120
mitre	2.054	10.724	39.704	115.155
mod008	6	319	1.243	307
mod010	146	2.655	11.203	6.548
p0201	133	201	1.923	7.615
seymour	4.944	1.372	33.549	423
stein27	118	27	378	18
stein45	331	45	1034	30

- 5 problemas binários da biblioteca MIPLIB 2003 <http://miplib.zib.de/> (Koch, Achterberg, Andersen, Bastert, Berthold, Bixby, Danna, Gamrath, Gleixner, Heinz, Lodi, Mittelman, Ralphs, Salvagnin, Steffy and Wolter 2011);
- 22 problemas binários da biblioteca MIPLIB 2010 <http://miplib.zib.de/> (Koch, Achterberg, Andersen, Bastert, Berthold, Bixby, Danna, Gamrath, Gleixner, Heinz, Lodi, Mittelman, Ralphs, Salvagnin, Steffy and Wolter 2011);

Uma descrição detalhada das instâncias, contendo o número de restrições, o número de valores não-zero em restrições, o número de variáveis binárias e a solução ótima, está disponível nas tabelas 4.1, 4.2 e 4.3.

Podemos perceber a diversidade das instâncias utilizadas, observando a variação em número de variáveis e número de restrições.

Tabela 4.2: Instâncias MIPLIB 2003

instância	restrições	var. binárias	não-zeros	solução ótima
disktom	399	10.000	30.000	-5.000
ds	656	67.732	1.024.059	93,52
fast0507	507	63.009	409.349	174,00
manna81	6.480	3.321	12.960	-13.164
protfold	2.112	1.835	23.491	-31

Tabela 4.3: Instâncias MIPLIB 2010

instância	restrições	var. binárias	não-zeros	valor ótimo
acc-tight5	3.052	1.339	16.134	0
ash608gpia-3col	24.748	3.651	74.244	Infeasible
bab5	4.964	21.600	155.520	-106.412
bley _x /l1	175.620	5.831	869.391	190
cov1075	637	120	14.280	20
eilB101	100	2.818	24.120	1.216
ex9	40.962	10.404	517.112	81
iss-100-0-cov	3.831	100	22.986	29
iss-bupa-cov	4.803	345	38.392	36
iss-pima-cov	7.201	768	71.941	33
m100n500k4r1	100	500	2.000	-25
macrophage	3.164	2.260	9.492	374
mine-166-5	8.429	830	19.412	-566.396.000
n3div36	4.484	22.120	340.740	130.800
neos18	11.402	3.312	24.614	16
netdiversion	119.589	129.180	615.282	242
ns1688347	4.191	2.685	66.908	27
opm2-z7-s2	31.798	2.023	79.762	-10.280
rmine6	7.078	1.096	18.084	-457
sp98ic	825	10.894	316.317	449.145.000
tanglegram1	68.342	34.759	205.026	5.183
vpphard	47.280	51.471	372.305	5

4.3 Experimentos e Resultados

A heurística proposta neste trabalho lê um série de instâncias-teste de problemas diversos (como descrito na seção anterior), a solução fracionária e uma solução inicial inteira viável para o problema. A solução inicial foi gerada utilizando a MIP heurística *Feasibility Pump* e informada para todas as heurísticas MIP testadas.

Para a maioria das aplicações de pesquisa práticas, métodos de soluções serão somente usuais se são capazes de produzir uma solução satisfatória para o problema em um curto período de tempo. Assim, foi utilizado o tempo limite de 300 segundos. O parâmetro tamanho inicial de problema a ser resolvido foi testado utilizando quatro valores : 25, 100, 200 e 250. Este tamanho refere-se ao número de variáveis livres que o problema deixa a cada execução. Outro parâmetro é o percentual de aumento no tamanho do problema, quando o tamanho atual não encontra melhores soluções, e que neste caso utilizamos 1,5. Para o método denominado pRINS2, foram testados 4 tipos de torneio: binário, ternário, quaternário e quinquenário. O torneio é utilizado neste método para definir a seleção das melhores variáveis a fixar. Para o algoritmo pRINS3, está definido o parâmetro *percentualShift* como sendo a quantidade de elementos que serão trocados na parte a ser fixada. Foram experimentados para este parâmetro os seguintes valores: 10%, 25% e 50%.

Os resultados descritos a seguir consideram a implementação do método RINS adaptado, em suas três versões: pRINS1, pRINS2 e pRINS3. As instâncias utilizadas nos testes são as descritas nas tabelas 4.1, 4.2 e 4.3.

São comparados nossos resultados com o resolvidor *CBC stand-alone*, e com uma implementação do método RINS na forma original. Ambos CBC e RINS usam a mesma solução inicial que nosso método. Para facilitar a leitura e melhorar o entendimento dos valores, estes foram divididos e estão apresentados nas tabelas 4.4, 4.5, 4.6, 4.7, 4.8, 4.9, 4.10, 4.11, 4.12.

Por fim, os dados são consolidados na Tabela 4.13. A métrica *gap* da solução, encontrada nas diversas tabelas, foi calculado de acordo com a seguinte expressão: $\min(100, (z - best) \div best \times 100)$, definindo a diferença entre o valor encontrado e o melhor valor (já descrito nas tabelas de caracterização das instâncias).

Na Tabela 4.4 são apresentadas as soluções encontradas para cada instância testada e o respectivo *gap*, para o método RINS e para o resolver *stand-alone* CBC.

A tabela 4.5, descreve o valores encontrados (solução e gap) para cada instância, considerando a implementação pRINS, variando o parâmetro tamanho inicial(T).

As tabelas 4.6, 4.7, 4.8, 4.9 apresentam os resultados da implementação pRINS2, com a variação nos métodos de torneio e no tamanho da solução inicial(T).

Em seguida, as tabelas 4.10, 4.11, 4.12 descrevem os valores encontrados para a variação pRINS3, considerando nos testes o parâmetro tamanho do problema inicial(T) e tamanho do shift(S).

Finalmente a Tabela 4.13 faz a comparação de todos os métodos, apresentando a soma e a média dos gaps para cada algoritmo e seus parâmetros.

4.4 Discussão dos Resultados

Considerando o método pRINS1, cujos resultados estão expressos na Tabela 4.5, o tamanho inicial que apresentou melhor desempenho foi 200 variáveis livres. Este resultado indicou que não foi eficiente iniciar a busca nos menores tamanhos, já que problemas muito fixados acabam perdendo diversidade.

Já para o método pRins2, em que os experimentos estão descritos nas tabelas 4.6, 4.7, 4.8 e 4.9, podemos perceber que a melhor combinação foi o torneio ternário, com tamanho inicial em 100. A técnica de torneios traz intensificação ao método, já que para um mesmo tamanho de fixações faremos a pesquisa em problemas semelhantes. Neste sentido o uso do torneio ternário se mostrou mais eficiente, pois em torneios quando mais elementos fazem parte do sorteio a tendência de escolher os melhores aumenta (torneio quaternário e quinquenário), diminuindo a diversificação. De outro modo contrário, o torneio binário tende a ser bastante aleatório, escolhendo elementos com prioridade de fixação baixa.

Para o método pRINS3, cujos testes são descritos nas tabelas 4.10, 4.11 e 4.12, a

melhor combinação de parâmetros encontrada foi *shift* de 10% e tamanho inicial em 25. O menor tamanho de *shift* permite maior número de tentativas de trocas, bem como desta maneira somente uma pequena porção de variáveis com baixa prioridade de fixação serão inseridas na lista de fixação. Com uma troca em 10% obtemos grande diversificação, já que podemos formar várias soluções semelhantes.

Por fim, na Tabela 4.13 podemos observar quais as combinações de parâmetros que obtiveram melhor desempenho quando comparadas ao método RINS original e ao resolvedor CBC *stand-alone*, assim como aquelas que não valem a pena ser utilizadas. O implementação pRINS2 Torneio Ternário $T = 100$ apresentou o melhor gap (526) e a melhor média (12,83).

Tabela 4.4: Resultados CBC, RINS

Instâncias	CBC		RINS	
	z	gap	z	gap
air03	340.160,00	0,00	561.226,00	64,98
air04	56.137,00	0,00	57.483,00	2,39
air05	26.374,00	0,00	29.795,00	12,97
cap6000	-2.451.377,00	0,00	-2.445.344,00	0,24
harp2	-73.728.304,00	0,23	-70.396.845,00	4,74
l152lav	4.722,00	0,58	4.785,00	0,73
lseu	1.120,00	0,00	1.243,00	10,98
mitre	115.155,00	0,00	115.155,00	0,00
mod008	307,00	0,00	307,00	0,00
mod010	6.548,00	0,00	6.548,00	0,00
p0201	7.615,00	0,00	8.510,00	11,75
seymour	447,00	5,67	431,00	1,89
stein27	18,00	0,00	19,00	5,55
stein45	30,00	0,00	33,00	10,00
disktom	-5.000,00	0,00	-5.000,00	0,00
ds	5.418,00	100,00	1.398,00	100,00
fast0507	184,00	5,74	176,00	1,14
manna81	-13.162,00	0,015	-13.162,00	0,015
protfold	-23,00	25,80	-23,00	25,80
acc-tight5	0,00	0,00	0,00	0
ash608gpia-3col	Infeasible	0,00	Infeasible	0,00
bab5	-71.020,00	33,25	-71.020,00	33,25
bley_xl1	235,00	23,68	205,00	7,89
cov1075	20,00	0,00	20,00	0,00
eilB101	1.289,00	6,00	1.477,00	21,46
ex9	81,00	0,00	81,00	0,00
iss-100-0-cov	29,00	0,00	29,00	0,00
iss-bupa-cov	39,00	8,33	39,00	8,33
iss-pima-cov	34,00	3,03	33,00	0,00
m100n500k4r1	-24,00	4,00	-21,00	16,00
macrophage	428,00	14,43	601,00	60,69
mine-166-5	-566.395.707,00	5,17	-565.431.610,00	0,17
n3div36	152.800,00	16,81	149.800,00	14,52
neos18	17,00	6,25	19,00	18,75
netdiversion	9.700.229,00	100,00	407,00	68,18
ns1688347	35,00	29,62	35,00	29,62
opm2-z7-s2	-8.216,00	20,07	-8.216,00	20,07
rmine6	-456,00	0,21	-456,00	0,21
sp98ic	458.815.551,00	2,15	464.766.964,00	3,47
tanglegram1	7.473,00	44,18	7.413,00	43,02
vpphard	22.813,00	100,00	22.813,00	100,00

Tabela 4.5: Resultados pRINS1

Instâncias	pRINS1							
	T=25		T=100		T=200		T=250	
	Z	gap	Z	gap	Z	gap	Z	gap
air03	340.160,00	0,00	340.160,00	0,00	340.160,00	0,00	340.160,00	0,00
air04	56.138,00	0,00	56.138,00	0,00	56.138,00	0,00	56.138,00	0,00
air05	26.374,00	0,00	26.374,00	0,00	26.374,00	0,00	26.374,00	0,00
cap6000	-2.445.340,00	0,25	-2.445.340,00	0,25	-2.445.340,00	0,25	-2.445.340,00	0,25
harp2	-62.462.500,00	15,48	-62.462.500,00	15,48	-62.462.500,00	15,48	-62.462.500,00	15,48
l152lav	4.722,00	0,59	4.732,00	0,38	4.732,00	0,38	4.732,00	0,38
lseu	1.120,00	0,00	1.120,00	0,00	1.120,00	0,00	1.120,00	0,00
mitre	115.155,00	0,00	115.155,00	0,00	115.155,00	0,00	115.155,00	0,00
mod008	307,00	0,00	307,00	0,00	307,00	0,00	307,00	0,00
mod010	6.548,00	0,00	6.548,00	0,00	6.548,00	0,00	6.548,00	0,00
p0201	7.815,00	2,63	7.675,00	0,79	7.675,00	0,79	7.675,00	0,79
seymour	429,00	1,42	429,00	1,42	429,00	1,42	429,00	1,42
stein27	18,00	0,00	18,00	0,00	18,00	0,00	18,00	0,00
stein45	33,00	10,00	33,00	10,00	33,00	10,00	33,00	10,00
disktom	-5.000,00	0,00	-5.000,00	0,00	-5.000,00	0,00	-5.000,00	0,00
ds	5.207,00	100,00	3.951,00	100,00	3.951,00	100,00	5.207,00	100,00
fast0507	187,00	7,47	185,00	6,32	185,00	6,32	187,00	7,47
manna81	-12.171,00	7,54	-12.171,00	7,54	-12.171,00	7,54	-12.171,00	7,54
protfold	-23,00	25,81	-23,00	25,81	-23,00	25,81	-23,00	25,81
acc-tight5	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
ash608gpia-3col	Infeasible	0,00	Infeasible	0,00	Infeasible	0,00	Infeasible	0,00
bab5	-71.020,00	33,26	-71.020,00	33,26	-71.020,00	33,26	-71.020,00	33,26
bley_xl1	230,00	21,05	205,00	7,89	215,00	13,16	215,00	13,16
cov1075	20,00	0,00	20,00	0,00	20,00	0,00	20,00	0,00
eilB101	1.619,00	33,14	1.619,00	33,14	1.619,00	33,14	1.619,00	33,14
ex9	81,00	0,00	81,00	0,00	81,00	0,00	81,00	0,00
iss-100-0-cov	30,00	3,45	30,00	3,45	30,00	3,45	30,00	3,45
iss-bupa-cov	37,00	2,78	39,00	8,33	37,00	2,78	38,00	5,56
iss-pima-cov	34,00	3,03	34,00	3,03	34,00	3,03	34,00	3,03
m100n500k4r1	-18,00	28,00	-18,00	28,00	-18,00	28,00	-18,00	28,00
macrophage	500,00	33,69	500,00	33,69	500,00	33,69	500,00	33,69
mine-166-5	-552.461.000,00	2,46	-552.461.000,00	2,46	-552.461.000,00	2,46	-552.461.000,00	2,46
n3div36	166.200,00	27,06	162.000,00	23,85	148.000,00	13,15	153.800,00	17,58
neos18	16,00	0,00	16,00	0,00	16,00	0,00	16,00	0,00
netdiversion	244,00	0,83	244,00	0,83	244,00	0,83	244,00	0,83
ns1688347	35,00	29,63	35,00	29,63	35,00	29,63	35,00	29,63
opm2-z7-s2	-8.216,00	20,08	-8.216,00	20,08	-8.216,00	20,08	-8.216,00	20,08
rmine6	-457,00	0,00	-457,00	0,00	-457,00	0,00	-457,00	0,00
sp98ic	472.122.202,00	5,12	471.906.775,00	5,07	471.906.775,00	5,07	471.906.775,00	5,07
tanglegram1	7.451,00	43,76	7.446,00	43,66	7.446,00	43,66	7.446,00	43,66
vpphard	6.191,00	100,00	6.197,00	100,00	6.197,00	100,00	6.197,00	100,00

Tabela 4.6: Resultados pRINS2, com Torneio Binário

Instâncias	pRINS2 Torneio Binário							
	T=25		T=100		T=200		T=250	
	Z	gap	Z	gap	Z	gap	Z	gap
air03	340.160,00	0,00	340.160,00	0,00	340.160,00	0,00	340.160,00	0,00
air04	56.212,00	0,13	56.212,00	0,13	56.212,00	0,13	56.212,00	0,13
air05	26.374,00	0,00	29.572,00	12,13	26.374,00	0,00	26.374,00	0,00
cap6000	-2.445.340,00	0,25	-2.445.340,00	0,25	-2.445.340,00	0,25	-2.445.340,00	0,25
harp2	-62.462.500,00	15,48	-62.462.500,00	15,48	-62.462.500,00	15,48	-62.462.500,00	15,48
l152lav	4.732,00	0,38	4.732,00	0,38	4.732,00	0,38	4.732,00	0,38
lseu	1.136,00	1,43	1.136,00	1,43	1.136,00	1,43	1.136,00	1,43
mitre	115.155,00	0,00	115.155,00	0,00	115.155,00	0,00	115.155,00	0,00
mod008	307,00	0,00	307,00	0,00	307,00	0,00	307,00	0,00
mod010	6.548,00	0,00	6.548,00	0,00	6.548,00	0,00	6.548,00	0,00
p0201	12.425,00	63,16	7.675,00	0,79	7.675,00	0,79	7.675,00	0,79
seymour	426,00	0,71	426,00	0,71	426,00	0,71	426,00	0,71
stein27	18,00	0,00	18,00	0,00	18,00	0,00	18,00	0,00
stein45	33,00	10,00	33,00	10,00	33,00	10,00	33,00	10,00
disktom	-5.000,00	0,00	-5.000,00	0,00	-5.000,00	0,00	-5.000,00	0,00
ds	5.207,00	100,00	5.207,00	100,00	5.207,00	100,00	5.207,00	100,00
fast0507	181,00	4,02	181,00	4,02	181,00	4,02	181,00	4,02
manna81	-12.171,00	7,54	-12.171,00	7,54	-12.171,00	7,54	-12.171,00	7,54
protfold	-23,00	25,81	-23,00	25,81	-23,00	25,81	-23,00	25,81
acc-tight5	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
ash608gpia-3col	Infeasible	0,00	Infeasible	0,00	Infeasible	0,00	Infeasible	0,00
bab5	-71.020,00	33,26	-71.020,00	33,26	-71.020,00	33,26	-71.020,00	33,26
bley_xl1	205,00	7,89	205,00	7,89	205,00	7,89	205,00	7,89
cov1075	20,00	0,00	20,00	0,00	20,00	0,00	20,00	0,00
eilB101	4.364,00	100,00	4.364,00	100,00	4.364,00	100,00	4.364,00	100,00
ex9	81,00	0,00	81,00	0,00	81,00	0,00	81,00	0,00
iss-100-0-cov	30,00	3,45	29,00	0,00	29,00	0,00	29,00	0,00
iss-bupa-cov	38,00	5,56	38,00	5,56	39,00	8,33	39,00	8,33
iss-pima-cov	34,00	3,03	34,00	3,03	34,00	3,03	34,00	3,03
m100n500k4r1	-18,00	28,00	-18,00	28,00	-18,00	28,00	-18,00	28,00
macrophage	469,00	25,40	469,00	25,40	469,00	25,40	475,00	27,01
mine-166-5	-552.461.000,00	2,46	-552.461.000,00	2,46	-552.461.000,00	2,46	-552.461.000,00	2,46
n3div36	163.400,00	24,92	166.200,00	27,06	166.200,00	27,06	166.200,00	27,06
neos18	20,00	25,00	20,00	25,00	20,00	25,00	20,00	25,00
netdiversion	244,00	0,83	244,00	0,83	244,00	0,83	244,00	0,83
ns1688347	35,00	29,63	35,00	29,63	35,00	29,63	35,00	29,63
opm2-z7-s2	-8.216,00	20,08	-8.216,00	20,08	-8.216,00	20,08	-8.216,00	20,08
rmine6	-457,00	0,00	-457,00	0,00	-457,00	0,00	-457,00	0,00
sp98ic	475.759.269,00	5,93	474.535.821,00	5,65	474.535.821,00	5,65	474.168.844,00	5,57
tanglegram1	7.446,00	43,66	7.451,00	43,76	7.451,00	43,76	7.451,00	43,76
vpphard	18.354,00	100,00	18.344,00	100,00	18.344,00	100,00	18.344,00	100,00

Tabela 4.7: Resultados pRINS2, com Torneio Ternário

Instâncias	pRINS2 Torneio Ternário							
	T=25		T=100		T=200		T=250	
	Z	gap	Z	gap	Z	gap	Z	gap
air03	340.160,00	0,00	340.160,00	0,00	340.160,00	0,00	340.160,00	0,00
air04	56.137,00	0,00	56.137,00	0,00	56.137,00	0,00	56.137,00	0,00
air05	26.374,00	0,00	26.374,00	0,00	26.374,00	0,00	26.374,00	0,00
cap6000	-2.445.340,00	0,25	-2.445.340,00	0,25	-2.445.340,00	0,25	-2.445.340,00	0,25
harp2	-62.462.500,00	15,48	-62.462.500,00	15,48	-62.462.500,00	15,48	-62.462.500,00	15,48
l152lav	4.722,00	0,59	4.722,00	0,59	4.722,00	0,59	4.722,00	0,59
lseu	1.120,00	0,00	1.120,00	0,00	1.120,00	0,00	1.120,00	0,00
mitre	115.155,00	0,00	115.155,00	0,00	115.155,00	0,00	115.155,00	0,00
mod008	307,00	0,00	307,00	0,00	307,00	0,00	307,00	0,00
mod010	6.548,00	0,00	6.548,00	0,00	6.548,00	0,00	6.548,00	0,00
p0201	12.375,00	62,51	7.675,00	0,79	7.675,00	0,79	7.675,00	0,79
seymour	426,00	0,71	426,00	0,71	426,00	0,71	426,00	0,71
stein27	18,00	0,00	18,00	0,00	18,00	0,00	18,00	0,00
stein45	33,00	10,00	33,00	10,00	33,00	10,00	33,00	10,00
disktom	-5.000,00	0,00	-5.000,00	0,00	-5.000,00	0,00	-5.000,00	0,00
ds	5.207,00	100,00	5.207,00	100,00	5.207,00	100,00	5.207,00	100,00
fast0507	183,00	5,17	183,00	5,17	183,00	5,17	183,00	5,17
manna81	-12.171,00	7,54	-12.171,00	7,54	-12.171,00	7,54	-12.171,00	7,54
protfold	-23,00	25,81	-23,00	25,81	-23,00	25,81	-23,00	25,81
acc-tight5	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
ash608gpia-3col	Infeasible	0,00	Infeasible	0,00	Infeasible	0,00	Infeasible	0,00
bab5	-71.020,00	33,26	-71.020,00	33,26	-71.020,00	33,26	-71.020,00	33,26
bley_xl1	205,00	7,89	205,00	7,89	205,00	7,89	205,00	7,89
cov1075	20,00	0,00	20,00	0,00	20,00	0,00	20,00	0,00
eilB101	1.477,00	21,46	1.477,00	21,46	1.477,00	21,46	1.477,00	21,46
ex9	81,00	0,00	81,00	0,00	81,00	0,00	81,00	0,00
iss-100-0-cov	30,00	3,45	30,00	3,45	31,00	6,90	31,00	6,90
iss-bupa-cov	38,00	5,56	38,00	5,56	39,00	8,33	39,00	8,33
iss-pima-cov	34,00	3,03	34,00	3,03	34,00	3,03	34,00	3,03
m100n500k4r1	-18,00	28,00	-18,00	28,00	-18,00	28,00	-18,00	28,00
macrophage	479,00	28,07	479,00	28,07	479,00	28,07	479,00	28,07
mine-166-5	-552.461.000,00	2,46	-552.461.000,00	2,46	-552.461.000,00	2,46	-552.461.000,00	2,46
n3div36	166.200,00	27,06	166.200,00	27,06	162.000,00	23,85	166.200,00	27,06
neos18	16,00	0,00	16,00	0,00	16,00	0,00	16,00	0,00
netdiversion	251,00	3,72	251,00	3,72	251,00	3,72	251,00	3,72
ns1688347	35,00	29,63	35,00	29,63	35,00	29,63	35,00	29,63
opm2-z7-s2	-8.216,00	20,08	-8.216,00	20,08	-8.216,00	20,08	-8.216,00	20,08
rmine6	-457,00	0,00	-457,00	0,00	-457,00	0,00	-457,00	0,00
sp98ic	463.439.059,00	3,18	463.439.059,00	3,18	463.439.059,00	3,18	463.439.059,00	3,18
tanglegram1	7.446,00	43,66	7.446,00	43,66	7.446,00	43,66	7.446,00	43,66
vpphard	18.344,00	100,00	18.344,00	100,00	18.344,00	100,00	18.344,00	100,00

Tabela 4.8: Resultados pRINS2, com Torneio Quartenário

Instâncias	pRINS2 Torneio Quartenário							
	T=25		T=100		T=200		T=250	
	Z	gap	Z	gap	Z	gap	Z	gap
air03	560.614,00	64,81	560.614,00	64,81	560.614,00	64,81	560.614,00	64,81
air04	56.489,00	0,63	56.347,00	0,37	56.489,00	0,63	56.489,00	0,63
air05	26.457,00	0,31	26.374,00	0,00	26.457,00	0,31	26.457,00	0,31
cap6000	-2.445.340,00	0,25	-2.445.340,00	0,25	-2.445.340,00	0,25	-2.445.340,00	0,25
harp2	-62.462.500,00	15,48	-62.462.500,00	15,48	-62.462.500,00	15,48	-62.462.500,00	15,48
l152lav	4.732,00	0,38	4.732,00	0,38	4.732,00	0,38	4.732,00	0,38
lseu	1.157,00	3,30	1.157,00	3,30	1.157,00	3,30	1.157,00	3,30
mitre	115.155,00	0,00	115.155,00	0,00	115.155,00	0,00	115.155,00	0,00
mod008	307,00	0,00	307,00	0,00	307,00	0,00	307,00	0,00
mod010	6.548,00	0,00	6.548,00	0,00	6.548,00	0,00	6.548,00	0,00
p0201	12.425,00	63,16	8.155,00	7,09	8.155,00	7,09	8.155,00	7,09
seymour	426,00	0,71	426,00	0,71	426,00	0,71	426,00	0,71
stein27	18,00	0,00	18,00	0,00	18,00	0,00	18,00	0,00
stein45	33,00	10,00	33,00	10,00	33,00	10,00	33,00	10,00
disktom	-5.000,00	0,00	-5.000,00	0,00	-5.000,00	0,00	-5.000,00	0,00
ds	5.207,00	100,00	5.207,00	100,00	5.207,00	100,00	5.207,00	100,00
fast0507	187,00	7,47	184,00	5,75	184,00	5,75	184,00	5,75
manna81	-12.171,00	7,54	-12.171,00	7,54	-12.171,00	7,54	-12.171,00	7,54
protfold	-23,00	25,81	-23,00	25,81	-23,00	25,81	-23,00	25,81
acc-tight5	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
ash608gpia-3col	Infeasible	0,00	Infeasible	0,00	Infeasible	0,00	Infeasible	0,00
bab5	-71.020,00	33,26	-71.020,00	33,26	-71.020,00	33,26	-71.020,00	33,26
bley_xl1	205,00	7,89	205,00	7,89	205,00	7,89	205,00	7,89
cov1075	20,00	0,00	20,00	0,00	20,00	0,00	20,00	0,00
eilB101	4.364,00	100,00	1.568,00	28,95	1.568,00	28,95	1.568,00	28,95
ex9	81,00	0,00	81,00	0,00	81,00	0,00	81,00	0,00
iss-100-0-cov	30,00	3,45	29,00	0,00	29,00	0,00	29,00	0,00
iss-bupa-cov	38,00	5,56	38,00	5,56	38,00	5,56	39,00	8,33
iss-pima-cov	34,00	3,03	34,00	3,03	34,00	3,03	34,00	3,03
m100n500k4r1	-18,00	28,00	-18,00	28,00	-18,00	28,00	-18,00	28,00
macrophage	485,00	29,68	485,00	29,68	485,00	29,68	485,00	29,68
mine-166-5	-552.461.000,00	2,46	-552.461.000,00	2,46	-552.461.000,00	2,46	-552.461.000,00	2,46
n3div36	156.200,00	19,42	156.200,00	19,42	156.200,00	19,42	156.200,00	19,42
neos18	20,00	25,00	20,00	25,00	20,00	25,00	20,00	25,00
netdiversion	317,00	30,99	317,00	30,99	317,00	30,99	317,00	30,99
ns1688347	35,00	29,63	35,00	29,63	35,00	29,63	35,00	29,63
opm2-z7-s2	-8.216,00	20,08	-8.216,00	20,08	-8.216,00	20,08	-8.216,00	20,08
rmine6	-457,00	0,00	-457,00	0,00	-457,00	0,00	-457,00	0,00
sp98ic	468.459.538,00	4,30	468.459.538,00	4,30	468.459.538,00	4,30	468.459.538,00	4,30
tanglegram1	7.446,00	43,66	7.446,00	43,66	7.446,00	43,66	7.446,00	43,66
vpphard	18.348,00	100,00	18.348,00	100,00	18.348,00	100,00	18.382,00	100,00

Tabela 4.9: Resultados pRINS2, com Torneio Quinquenário

Instâncias	pRINS2 Torneio Quinquenário							
	T=25		T=100		T=200		T=250	
	Z	gap	Z	gap	Z	gap	Z	gap
air03	340.160,00	0,00	340.160,00	0,00	340.160,00	0,00	340.160,00	0,00
air04	56.438,00	0,54	56.438,00	0,54	56.438,00	0,54	56.438,00	0,54
air05	26.374,00	0,00	26.448,00	0,28	26.469,00	0,36	26.457,00	0,31
cap6000	-2.445.340,00	0,25	-2.445.340,00	0,25	-2.445.340,00	0,25	-2.445.340,00	0,25
harp2	-62.462.500,00	15,48	-62.462.500,00	15,48	-62.462.500,00	15,48	-62.462.500,00	15,48
l152lav	4.722,00	0,59	4.732,00	0,38	4.732,00	0,38	4.732,00	0,38
lseu	1.120,00	0,00	1.120,00	0,00	1.120,00	0,00	1.120,00	0,00
mitre	115.155,00	0,00	115.155,00	0,00	115.155,00	0,00	115.155,00	0,00
mod008	307,00	0,00	307,00	0,00	307,00	0,00	307,00	0,00
mod010	6.548,00	0,00	6.548,00	0,00	6.548,00	0,00	6.548,00	0,00
p0201	7.815,00	2,63	8.155,00	7,09	7.675,00	0,79	7.675,00	0,79
seymour	427,00	0,95	427,00	0,95	427,00	0,95	427,00	0,95
stein27	18,00	0,00	18,00	0,00	18,00	0,00	18,00	0,00
stein45	33,00	10,00	33,00	10,00	33,00	10,00	33,00	10,00
disktom	-5.000,00	0,00	-5.000,00	0,00	-5.000,00	0,00	-5.000,00	0,00
ds	5.207,00	100,00	5.207,00	100,00	5.207,00	100,00	5.207,00	100,00
fast0507	184,00	5,75	184,00	5,75	184,00	5,75	184,00	5,75
manna81	-12.171,00	7,54	-12.171,00	7,54	-12.171,00	7,54	-12.171,00	7,54
protfold	-23,00	25,81	-23,00	25,81	-23,00	25,81	-23,00	25,81
acc-tight5	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
ash608gpia-3col	Infeasible	0,00	Infeasible	0,00	Infeasible	0,00	Infeasible	0,00
bab5	-71.020,00	33,26	-71.020,00	33,26	-71.020,00	33,26	-71.020,00	33,26
bley_xl1	205,00	7,89	205,00	7,89	205,00	7,89	205,00	7,89
cov1075	20,00	0,00	20,00	0,00	20,00	0,00	20,00	0,00
eilB101	1.692,00	39,14	1.692,00	39,14	1.692,00	39,14	1.598,00	31,41
ex9	81,00	0,00	81,00	0,00	81,00	0,00	81,00	0,00
iss-100-0-cov	30,00	3,45	30,00	3,45	30,00	3,45	30,00	3,45
iss-bupa-cov	37,00	2,78	37,00	2,78	37,00	2,78	37,00	2,78
iss-pima-cov	34,00	3,03	34,00	3,03	34,00	3,03	34,00	3,03
m100n500k4r1	-18,00	28,00	-18,00	28,00	-18,00	28,00	-18,00	28,00
macrophage	490,00	31,02	490,00	31,02	490,00	31,02	490,00	31,02
mine-166-5	-552.461.000,00	2,46	-552.461.000,00	2,46	-552.461.000,00	2,46	-552.461.000,00	2,46
n3div36	159.200,00	21,71	162.000,00	23,85	162.000,00	23,85	159.200,00	21,71
neos18	20,00	25,00	20,00	25,00	20,00	25,00	20,00	25,00
netdiversion	244,00	0,83	248,00	2,48	248,00	2,48	248,00	2,48
ns1688347	35,00	29,63	35,00	29,63	35,00	29,63	35,00	29,63
opm2-z7-s2	-8.216,00	20,08	-8.216,00	20,08	-8.216,00	20,08	-8.216,00	20,08
rmine6	-457,00	0,00	-457,00	0,00	-457,00	0,00	-457,00	0,00
sp98ic	469.520.727,00	4,54	470.506.348,00	4,76	470.506.348,00	4,76	470.506.348,00	4,76
tanglegram1	7.446,00	43,66	7.446,00	43,66	7.446,00	43,66	7.446,00	43,66
vpphard	18.344,00	100,00	18.344,00	100,00	18.344,00	100,00	18.344,00	100,00

Tabela 4.10: Resultados pRINS3, '10%' Shift

Instâncias	pRINS3 : S = 10							
	T=25		T=100		T=200		T=250	
	Z	gap	Z	gap	Z	gap	Z	gap
air03	340.160,00	0,00	340.160,00	0,00	340.160,00	0,00	340.160,00	0,00
air04	56.138,00	0,00	56.622,00	0,86	56.369,00	0,41	56.335,00	0,35
air05	26.374,00	0,00	26.733,00	1,36	27.137,00	2,89	26.469,00	0,36
cap6000	-2.445.340,00	0,25	-2.445.340,00	0,25	-2.445.340,00	0,25	-2.445.340,00	0,25
harp2	-62.462.500,00	15,48	-62.462.500,00	15,48	-62.462.500,00	15,48	-62.462.500,00	15,48
l152lav	4.722,00	0,59	4.722,00	0,59	4.722,00	0,59	4.722,00	0,59
lseu	1.120,00	0,00	1.120,00	0,00	1.120,00	0,00	1.120,00	0,00
mitre	115.155,00	0,00	115.155,00	0,00	115.155,00	0,00	115.155,00	0,00
mod008	307,00	0,00	307,00	0,00	307,00	0,00	307,00	0,00
mod010	6.548,00	0,00	6.548,00	0,00	6.548,00	0,00	6.548,00	0,00
p0201	7.665,00	0,66	7.665,00	0,66	7.665,00	0,66	7.665,00	0,66
seymour	430,00	1,65	430,00	1,65	430,00	1,65	430,00	1,65
stein27	18,00	0,00	18,00	0,00	18,00	0,00	18,00	0,00
stein45	33,00	10,00	33,00	10,00	33,00	10,00	33,00	10,00
disktom	-5.000,00	0,00	-5.000,00	0,00	-5.000,00	0,00	-5.000,00	0,00
ds	5.207,00	100,00	5.207,00	100,00	5.207,00	100,00	5.207,00	100,00
fast0507	187,00	7,47	186,00	6,90	186,00	6,90	186,00	6,90
manna81	-12.171,00	7,54	-12.171,00	7,54	-12.171,00	7,54	-12.171,00	7,54
protfold	-23,00	25,81	-23,00	25,81	-23,00	25,81	-23,00	25,81
acc-tight5	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
ash608gpia-3col	Infeasible	0,00	Infeasible	0,00	Infeasible	0,00	Infeasible	0,00
bab5	-71.020,00	33,26	-71.020,00	33,26	-71.020,00	33,26	-71.020,00	33,26
bley_xl1	205,00	7,89	205,00	7,89	205,00	7,89	205,00	7,89
cov1075	20,00	0,00	20,00	0,00	20,00	0,00	20,00	0,00
eilB101	1.715,00	41,04	1.715,00	41,04	1.715,00	41,04	1.715,00	41,04
ex9	81,00	0,00	81,00	0,00	81,00	0,00	81,00	0,00
iss-100-0-cov	33,00	13,79	32,00	10,34	31,00	6,90	31,00	6,90
iss-bupa-cov	38,00	5,56	40,00	11,11	39,00	8,33	39,00	8,33
iss-pima-cov	34,00	3,03	34,00	3,03	34,00	3,03	34,00	3,03
m100n500k4r1	-18,00	28,00	-18,00	28,00	-18,00	28,00	-18,00	28,00
macrophage	505,00	35,03	505,00	35,03	505,00	35,03	505,00	35,03
mine-166-5	-552.461.000,00	2,46	-552.461.000,00	2,46	-552.461.000,00	2,46	-552.461.000,00	2,46
n3div36	135.600,00	3,67	166.220,00	27,08	162.000,00	23,85	166.220,00	27,08
neos18	18,00	12,50	18,00	12,50	18,00	12,50	18,00	12,50
netdiversion	244,00	0,83	244,00	0,83	244,00	0,83	251,00	3,72
ns1688347	35,00	29,63	35,00	29,63	35,00	29,63	34,00	25,93
opm2-z7-s2	-8.216,00	20,08	-8.216,00	20,08	-8.216,00	20,08	-8.216,00	20,08
rmine6	-457,00	0,00	-457,00	0,00	-457,00	0,00	-457,00	0,00
sp98ic	465.577.381,00	3,66	464.177.659,00	3,35	465.604.313,00	3,66	475.128.490,00	5,79
tanglegram1	7.470,00	44,13	7.470,00	44,13	7.470,00	44,13	7.470,00	44,13
vpphard	18.382,00	100,00	18.382,00	100,00	18.382,00	100,00	18.348,00	100,00

Tabela 4.11: Resultados pRINS3, '25%' Shift

Instâncias	pRINS3 : S = 25							
	T=25		T=100		T=200		T=250	
	Z	gap	Z	gap	Z	gap	Z	gap
air03	340.160,00	0,00	340.160,00	0,00	340.160,00	0,00	340.160,00	0,00
air04	56.758,00	1,11	56.758,00	1,11	56.758,00	1,11	56.777,00	1,14
air05	26.374,00	0,00	26.374,00	0,00	26.374,00	0,00	27.284,00	3,45
cap6000	-2.445.340,00	0,25	-2.445.340,00	0,25	-2.445.340,00	0,25	-2.445.340,00	0,25
harp2	-62.462.500,00	15,48	-62.462.500,00	15,48	-62.462.500,00	15,48	-62.462.500,00	15,48
l152lav	4.722,00	0,59	4.732,00	0,38	4.732,00	0,38	4.732,00	0,38
lseu	1.128,00	0,71	1.128,00	0,71	1.128,00	0,71	1.128,00	0,71
mitre	115.155,00	0,00	115.155,00	0,00	115.155,00	0,00	115.155,00	0,00
mod008	307,00	0,00	307,00	0,00	307,00	0,00	307,00	0,00
mod010	6.548,00	0,00	6.548,00	0,00	6.548,00	0,00	6.548,00	0,00
p0201	7.665,00	0,66	7.665,00	0,66	7.665,00	0,66	7.665,00	0,66
seymour	429,00	1,42	429,00	1,42	429,00	1,42	429,00	1,42
stein27	18,00	0,00	18,00	0,00	18,00	0,00	18,00	0,00
stein45	33,00	10,00	33,00	10,00	33,00	10,00	33,00	10,00
disktom	-5.000,00	0,00	-5.000,00	0,00	-5.000,00	0,00	-5.000,00	0,00
ds	5.207,00	100,00	5.207,00	100,00	5.207,00	100,00	5.207,00	100,00
fast0507	180,00	3,45	181,00	4,02	180,00	3,45	182,00	4,60
manna81	-12.171,00	7,54	-12.171,00	7,54	-12.171,00	7,54	-12.171,00	7,54
protfold	-23,00	25,81	-23,00	25,81	-23,00	25,81	-23,00	25,81
acc-tight5	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
ash608gpia-3col	Infeasible	0,00	Infeasible	0,00	Infeasible	0,00	Infeasible	0,00
bab5	-71.020,00	33,26	-71.020,00	33,26	-71.020,00	33,26	-71.020,00	33,26
bley_xl1	205,00	7,89	205,00	7,89	205,00	7,89	205,00	7,89
cov1075	20,00	0,00	20,00	0,00	20,00	0,00	20,00	0,00
eilB101	4.364,00	100,00	4.364,00	100,00	4.364,00	100,00	4.364,00	100,00
ex9	81,00	0,00	81,00	0,00	81,00	0,00	81,00	0,00
iss-100-0-cov	32,00	10,34	30,00	3,45	30,00	3,45	30,00	3,45
iss-bupa-cov	40,00	11,11	40,00	11,11	39,00	8,33	39,00	8,33
iss-pima-cov	34,00	3,03	34,00	3,03	34,00	3,03	34,00	3,03
m100n500k4r1	-18,00	28,00	-18,00	28,00	-18,00	28,00	-18,00	28,00
macrophage	505,00	35,03	505,00	35,03	505,00	35,03	505,00	35,03
mine-166-5	-552.461.000,00	2,46	-552.461.000,00	2,46	-552.461.000,00	2,46	-552.461.000,00	2,46
n3div36	166.220,00	27,08	166.220,00	27,08	166.220,00	27,08	166.220,00	27,08
neos18	16,00	0,00	16,00	0,00	16,00	0,00	16,00	0,00
netdiversion	251,00	3,72	251,00	3,72	251,00	3,72	251,00	3,72
ns1688347	34,00	25,93	35,00	29,63	34,00	25,93	34,00	25,93
opm2-z7-s2	-8.216,00	20,08	-8.216,00	20,08	-8.216,00	20,08	-8.216,00	20,08
rmine6	-457,00	0,00	-457,00	0,00	-457,00	0,00	-457,00	0,00
sp98ic	509.915.918,00	13,53	526.200.292,00	17,16	509.915.918,00	13,53	509.915.918,00	13,53
tanglegram1	7.470,00	44,13	7.470,00	44,13	7.470,00	44,13	7.470,00	44,13
vpphard	18.633,00	100,00	18.348,00	100,00	18.382,00	100,00	18.348,00	100,00

Tabela 4.12: Resultados pRINS3, '50%' Shift

Instâncias	pRINS3 : S = 50							
	T=25		T=100		T=200		T=250	
	Z	gap	Z	gap	Z	gap	Z	gap
air03	340.160,00	0,00	340.160,00	0,00	340.160,00	0,00	340.160,00	0,00
air04	56.151,00	0,02	56.151,00	0,02	56.151,00	0,02	56.151,00	0,02
air05	26.469,00	0,36	26.513,00	0,53	26.469,00	0,36	26.374,00	0,00
cap6000	-2.445.340,00	0,25	-2.445.340,00	0,25	-2.445.340,00	0,25	-2.445.340,00	0,25
harp2	-62.462.500,00	15,48	-62.462.500,00	15,48	-62.462.500,00	15,48	-62.462.500,00	15,48
l152lav	4.722,00	0,59	4.722,00	0,59	4.722,00	0,59	4.722,00	0,59
lseu	1.120,00	0,00	1.120,00	0,00	1.120,00	0,00	1.120,00	0,00
mitre	115.155,00	0,00	115.155,00	0,00	115.155,00	0,00	115.155,00	0,00
mod008	307,00	0,00	307,00	0,00	307,00	0,00	307,00	0,00
mod010	6.548,00	0,00	6.548,00	0,00	6.548,00	0,00	6.548,00	0,00
p0201	8.215,00	7,88	7.815,00	2,63	7.818,00	2,67	7.818,00	2,67
seymour	427,00	0,95	427,00	0,95	427,00	0,95	427,00	0,95
stein27	18,00	0,00	18,00	0,00	18,00	0,00	18,00	0,00
stein45	33,00	10,00	33,00	10,00	33,00	10,00	33,00	10,00
disktom	-5.000,00	0,00	-5.000,00	0,00	-5.000,00	0,00	-5.000,00	0,00
ds	5.207,00	100,00	5.207,00	100,00	5.207,00	100,00	5.207,00	100,00
fast0507	182,00	4,60	181,00	4,02	181,00	4,02	181,00	4,02
manna81	-12.171,00	7,54	-12.171,00	7,54	-12.171,00	7,54	-12.171,00	7,54
protfold	-23,00	25,81	-23,00	25,81	-23,00	25,81	-23,00	25,81
acc-tight5	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
ash608gpia-3col	Infeasible	0,00	Infeasible	0,00	Infeasible	0,00	Infeasible	0,00
bab5	-71.020,00	33,26	-71.020,00	33,26	-71.020,00	33,26	-71.020,00	33,26
bley_xl1	205,00	7,89	205,00	7,89	205,00	7,89	205,00	7,89
cov1075	37,00	85,00	37,00	85,00	37,00	85,00	37,00	85,00
eilB101	4.364,00	100,00	4.364,00	100,00	3.170,00	100,00	4.364,00	100,00
ex9	81,00	0,00	81,00	0,00	81,00	0,00	81,00	0,00
iss-100-0-cov	30,00	3,45	30,00	3,45	30,00	3,45	30,00	3,45
iss-bupa-cov	40,00	11,11	40,00	11,11	39,00	8,33	39,00	8,33
iss-pima-cov	34,00	3,03	34,00	3,03	34,00	3,03	34,00	3,03
m100n500k4r1	-18,00	28,00	-18,00	28,00	-18,00	28,00	-18,00	28,00
macrophage	505,00	35,03	505,00	35,03	505,00	35,03	505,00	35,03
mine-166-5	-552.461.000,00	2,46	-552.461.000,00	2,46	-552.461.000,00	2,46	-552.461.000,00	2,46
n3div36	166.220,00	27,08	166.220,00	27,08	166.220,00	27,08	161.600,00	23,55
neos18	16,00	0,00	16,00	0,00	16,00	0,00	16,00	0,00
netdiversion	251,00	3,72	251,00	3,72	251,00	3,72	251,00	3,72
ns1688347	34,00	25,93	35,00	29,63	34,00	25,93	34,00	25,93
opm2-z7-s2	-8.216,00	20,08	-8.216,00	20,08	-8.216,00	20,08	-8.216,00	20,08
rmine6	-457,00	0,00	-457,00	0,00	-457,00	0,00	-457,00	0,00
sp98ic	493.434.241,00	9,86	502.684.024,00	11,92	493.434.241,00	9,86	493.434.241,00	9,86
tanglegram1	7.470,00	44,13	7.470,00	44,13	7.470,00	44,13	7.470,00	44,13
vpphard	18.348,00	100,00	18.382,00	100,00	18.344,00	100,00	18.348,00	100,00

Tabela 4.13: Resultados Consolidados: Soma e média dos gaps

Método	Soma Gaps	Média Gaps
CBC	550,00	13,41
RINS	698,00	17,02
pRINS1 T = 25	558,00	13,61
pRINS1 T = 100	544,00	13,27
pRINS1 T = 200	533,00	13,00
pRINS1 T = 250	541,00	13,20
pRINS2 Torneio Binário T = 25	688,00	16,78
pRINS2 Torneio Binário T = 100	636,00	15,51
pRINS2 Torneio Binário T = 200	626,00	15,27
pRINS2 Torneio Binário T = 250	628,00	15,32
pRINS2 Torneio Ternário T = 25	588,00	14,34
pRINS2 Torneio Ternário T = 100	526,00	12,83
pRINS2 Torneio Ternário T = 200	529,00	12,90
pRINS2 Torneio Ternário T = 250	533,00	13,00
pRINS2 Torneio Quartenário T = 25	786,00	19,17
pRINS2 Torneio Quartenário T = 100	653,00	15,93
pRINS2 Torneio Quartenário T = 200	653,00	15,93
pRINS2 Torneio Quartenário T = 250	656,00	16,00
pRINS2 Torneio Quinquenário T = 25	565,00	13,78
pRINS2 Torneio Quinquenário T = 100	574,00	14,00
pRINS2 Torneio Quinquenário T = 200	568,00	13,85
pRINS2 Torneio Quinquenário T = 250	558,00	13,61
pRINS3 Shift 10 T = 25	553,00	13,49
pRINS3 Shift 10 T = 100	580,00	14,15
pRINS3 Shift 10 T = 200	572,00	13,95
pRINS3 Shift 10 T = 250	574,00	14,00
pRINS3 Shift 25 T = 25	632,00	15,41
pRINS3 Shift 25 T = 100	633,00	15,44
pRINS3 Shift 25 T = 200	622,00	15,17
pRINS3 Shift 25 T = 250	627,00	15,29
pRINS3 Shift 50 T = 25	713,00	17,39
pRINS3 Shift 50 T = 100	713,00	17,39
pRINS3 Shift 50 T = 200	704,00	17,17
pRINS3 Shift 50 T = 250	701,00	17,10

Capítulo 5

Considerações Finais e Trabalhos Futuros

Este trabalho apresentou um método para resolver problemas binários genéricos. O método proposto, denominado pRINS, é uma heurística de programação inteira mista (*MIP heuristic* ou *matheuristic*) baseada no método *Relaxation Induced Neighborhood Search* (RINS).

Ao contrário do método RINS, o método pRINS utiliza técnicas de pré-processamento e busca binária visando a encontrar o número ideal de fixações em valores binários, dado um tamanho de problema a ser resolvido. Encontrado este número de fixações, utiliza-se um vetor de prioridades para definir quais as variáveis com a maior probabilidade de serem escolhidas para fixação. Esta escolha se dá de três critérios diferentes: 1) sequencialmente; 2) seleção por torneio e 3) por trocas (*shifts*). No primeiro critério, as variáveis são escolhidas na ordem do vetor, o qual é ordenado de forma crescente pela diferença, em módulo, entre os valores da solução inteira e fracionária. No segundo critério, há quatro possibilidade de realizar o torneio: binário, ternário, quaternário e quinquenário. Em cada um deles, são sorteados aleatoriamente um número de variáveis do vetor, por exemplo, se o torneio for ternário, são escolhidas três variáveis, e aquela com menor prioridade é a escolhida para ser fixada. No terceiro critério, é definido inicialmente um tamanho de troca, isto é, uma quantidade de variáveis do vetor de prioridades a serem trocadas com igual número de variáveis além do número de fixações definido preliminarmente. Para exemplificar, se o número de fixações preliminarmente definido foi de 50 variáveis e o tamanho de troca for de 5 variáveis (10% de 50), então as 5 primeiras variáveis do vetor de prioridade são trocadas com as 5 primeiras variáveis

após a quinquagésima variável, isto é, as variáveis 51 a 55. A seguir, resolvem-se os sub-MIPs resultantes desta troca. De forma iterativa, trocam-se, agora, o segundo conjunto de variáveis do vetor de prioridade (as variáveis 6 a 10), com as variáveis 51 a 55, e resolvem-se os sub-MIPs resultantes. O processo continua até que sejam trocadas as variáveis 46 a 50 com as variáveis 51 a 55. Se durante este processo uma nova solução for encontrada, a solução corrente é atualizada.

Cada critério de escolha de variáveis para fixação deu origem a uma versão do método pRINS. Na variante pRINS1, o critério de escolha é o sequencial; na variante pRINS2 a seleção é por torneio e na variante pRINS3, a escolha é pelo método de trocas.

Para testar a matheurística proposta foram utilizadas instâncias envolvendo problemas binários genéricos largamente utilizados na literatura.

Para cada variante foram testados diversos parâmetros. Para a variante pRINS1 foram testados os parâmetros tamanho inicial do problema desejado, variando entre 25, 100, 200 e 250 variáveis livres. Para a variante pRINS2 os parâmetros testados foram os seguintes: tipo de torneio (binário, ternário, quaternário e quinquenário) e o tamanho inicial do problema, com as mesmas variações apontadas acima. Para a variante pRINS3, foram testados os parâmetros tamanho do percentual de trocas, assim como o tamanho inicial do problema. Os percentuais de troca foram: 10%, 25% e 50%, enquanto o tamanho inicial do problema foi o mesmo dos critérios anteriores.

Os resultados computacionais mostraram que as variantes do método pRINS encontraram diversas soluções melhores ou iguais aos demais (usando somente CBC ou RINS original), na maioria dos casos. O ajuste de parâmetros procurou encontrar e entender quais as combinações destes geram os melhores resultados. Além disso o algoritmo foi capaz de produzir boas soluções viáveis em tempo computacional reduzido.

De acordo com os resultados encontrados, a variante pRINS2, com os parâmetros torneio ternário e tamanho inicial igual a 100, foi a que produziu os melhores resultados para os conjuntos de instâncias testadas. Por melhores resultados, entende-se a menor soma dos gaps para os conjuntos de instâncias.

Como extensão deste trabalho pode-se propor primeiramente um aprimoramento no modo como o vetor de prioridade é calculado, inserindo por exemplo uma métrica relacionada ao impacto da fixação de uma variável em relação às outras variáveis. Também um importante trabalho futuro é permitir que o algoritmo seja executado em diversos nós da árvore. Outra possibilidade interessante é a ampliação do conjunto de instâncias

para problemas não somente binários, mas que envolvam variáveis inteiras e variáveis mistas.

Apêndice A

Apêndices

A.1 Publicações

Neste apêndice são listados os trabalhos desenvolvidos nesta pesquisa que foram publicados em periódicos ou apresentados em eventos científicos até esta data.

- Santos, H. G., Souza, M. J. F., Gomes, T.M (2013). A pre-processing aware RINS based MIP heuristic. Proceedings of the 8th International Workshop on Hybrid Metaheuristics (HM '13), Ischia Island (Napoly), Maio de 2013, Itália. Lecture Notes in Computer Science. Springer Berlin Heidelberg, v. 7919, p. 1-11, 2013.
- Santos, H. G., Souza, M. J. F., Gomes, T.M (2013). Uma adaptação da heurística RINS aplicada a problemas binários utilizando resolvidor CBC *XLV Simpósio Brasileiro de Pesquisa Operacional* (SBPO '13), Rio de Janeiro, 2013.

Referências Bibliográficas

- Achterberg, T. and Berthold, T.: 2007, Improving the feasibility pump, *Discrete Optimization* **4**, 77–86.
- Baena, D. and Castro, J.: 2011, Using the analytic center in the feasibility pump, *Operations Research Letters* **29**, 310–317.
- Berthold, T.: 2006, *Primal heuristics for mixed integer programs*, Master’s thesis, Technische Universit at Berlin.
- Berthold, T.: 2008, Rens - relaxation enforced neighborhood search, *ZIB-Report* .
- Blum, C., Puchingerb, J., Raidl, G. and Roli., A.: 2011, Hybrid metaheuristics in combinatorial optimization: A survey applied soft computing, **11**, 4135–4151.
- Bonami, P., Cornuéjols, G., Lodi, A. and Margot, F.: 2009, A feasibility pump for mixed integer nonlinear programs, *Math. Program* **119**, 331–352.
- COIN-OR Branch-and-Cut MIP Solver*. 2014.
- Danna, E., Rothberg, E. and Le Pape, C.: 2003, Exploring relaxation induced neighborhoods to improve mip solutions, *Technical report*, ILOG.
- Dantzig, G.: 1951, Maximization of a linear function of variables subject to linear inequalities., *Activity Analysis of Production and Allocation. KOOPMANS,C (Ed.)*. New York: Wiley pp. 359–373.
- Eckstein, J. and Nediak, M.: 2007, Pivot, cut, and dive: a heuristic for 0-1 mixed integer programming, *Journal Heuristics* **13**, 471–503.
- Fampa, M. and Maculan, N.: 2006, *Otimização Linear*, 1 edn, chapter Rápido histórico e introdução, pp. 11–13.

- Fischetti, M., Glover, F. and Lodi, A.: 2005, The feasibility pump, *Mathematical Programming* **104**.
- Fischetti, M. and Lodi, A.: 2003, Local branching, *Mathematical Programming* **98**, 23–47.
- Fischetti, M. and Monaci, M.: 2012, Proximity search for 0-1 mixed-integer convex programming, *Technical Report, DEI, University of Padova (in preparation)* .
- Fischetti, M., Polo, C. and Scantamburlo, M.: 2004, A local branching heuristic for mixed-integer programs with 2-level variables, *Networks* **44**(2), 61–72.
- Fischetti, M. and Salvagnin, D.: 2009, Feasibility pump 2.0, *Math. Prog. Comp* **1**, 201â222.
- Forrest, J. and Lougee-Heimer, R.: 2005, *INFORMS Tutorials in Operations Research. p.257-277. CBC User Guide.*
- Fourier, J.: 1890, Oeuvres. ‘second extrait’, *G. DARBOUX, Gauthiers-Villar* pp. 325–328.
- Ge, D., He, S., Ye, Y. and Zhang, J.: 2011, Geometric rounding: a dependent randomized rounding scheme, *Journal of Combinatorial Optimization* **22**, 699–725.
- Glover, F.: 1986, Future paths for integer programming and artificial intelligence, *Computers & Operations Research* **13**, 533–549.
- Glover, F.: 1997, Tabu search and adaptive memory programming - advances, applications and challenges, in R. Barr, R. Helgason and J. Kennington (eds), *Advances in Metaheuristics, Optimization and Stochastic Modeling Technologies*, Kluwer, Boston, M.A., pp. 1–75.
- GLPK - GNU Project*: 2014.
- Goldberg, D.: 1989, *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley Professional; 1 edition (January 11, 1989).
- Hansen, P. and Mladenović, N.: 1997, Variable neighborhood search, *Computers & Operations Research* **24**(11), 1097–1100.
- IBM CPLEX Optimizer*: 2014.

- Kirkpatrick, S., Gelatt, C. and Vecchi, M.: 1983, Optimization by simulated annealing, *Science* **220**, 671–680.
- Koch, T., Achterberg, T., Andersen, E., Bastert, O., Berthold, T., Bixby, R., Danna, E., Gamrath, G., Gleixner, A., Heinz, S., Lodi, A., Mittelman, H., Ralphs, T., Salvagnin, D., Steffy, D. and Wolter, K.: 2011, Miplib 2010, *Mathematical Programming Computation* **3**, 103–163. 10.1007/s12532-011-0025-9.
- Land, A. and Doig, A.: 1960, An automatic method of solving discrete programming problems, *Econometrica* **28** **3**, 497–520.
- Lougee-Heimer, R.: 2003, The common optimization interface for operations research: Promoting open-source software in the operations research community, *IBM Journal of Research and Development* **47**, 57–66.
- lp solve reference guide*: 2014.
- Melo, T., Nickel, S. and da Gama, F. S.: 2009, An lp-rounding heuristic to solve a multi-period facility relocation problem, *Berichte des Fraunhofer ITWM* **168**, 1–37.
- Miller, J. and Franz, L.: 1996, A binary-rounding heuristic for multi-period variable-task-duration assignment problems, *Computers Ops Res* **23**, 819–828.
- MINTO - Mixed INTeger Optimizer*: 2014.
- Mixed Integer Linear Programming Benchmark*: 2011.
- Mixed Integer Linear Programming Benchmark*: 2013.
- Paquete, L.: 2012, *Manual de Computação Evolutiva e Metaheurística*, Vol. 2, 1 edn, Imprensa da Universidade de Coimbra, chapter Pesquisa Local Iterativa e em Vizinhança Variável, pp. 235–246.
- Reeves, C.: 1993, Genetic algorithms, *Modern Heuristic Techniques for Combinatorial Problems*, Advanced Topics in Computer Science Series, Blackwell Scientific Publications, pp. 151–196.
- Resende, M., Mateus, G. and Silva, R.: 2012, *Manual de Computação Evolutiva e Metaheurística*, Vol. 2, 1 edn, Imprensa da Universidade de Coimbra, chapter GRASP, pp. 203–212.
- Rothberg, E.: n.d., An evolutionary algorithm for polishing mixed integer programming solutions, *INFORMS Journal on Computing* **19**, 534–541.

Santos, H. G.: 2011, *CBC COIN-OR Branch-and-Cut - Short Guide to the Command Line Interface*.

Solving Constraint Integer Programs: 2014.

Souza, M.: 2012, *Manual de Computação Evolutiva e Metaheurística*, Vol. 2, 1 edn, Imprensa da Universidade de Coimbra, chapter Busca Tabu, pp. 177–202.

Souza, M.: 2014, Inteligência computacional para otimização.

SYMPHONY: 2014.

Wallace, C.: 2010, Zi round, a mip rounding heuristic, *Journal Heuristics* **16**, 715–722.