UNIVERSIDADE FEDERAL DE SÃO CARLOS

CENTRO DE CIÊNCIAS EXATAS E DE TECNOLOGIA

PROGRAMA DE PÓS-GRADUAÇÃO EM

CIÊNCIA DA COMPUTAÇÃO

UMA ESTRUTURA DE VIZINHANÇA BASEADA EM ÁRVORE DE COBERTURA APLICADA EM UMA COLABORAÇÃO DE ALGORITMO GENÉTICO E VNS PARA A MINIMIZAÇÃO DE MAKESPAN EM PROBLEMAS DE PROGRAMAÇÃO REATIVA DA PRODUÇÃO

CARLOS CESAR MANSUR TUMA

ORIENTADOR: PROF. DR. ORIDES MORANDIN JUNIOR

São Carlos - SP

Março/2015

UNIVERSIDADE FEDERAL DE SÃO CARLOS

CENTRO DE CIÊNCIAS EXATAS E DE TECNOLOGIA

PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

UMA ESTRUTURA DE VIZINHANÇA BASEADA EM ÁRVORE DE COBERTURA APLICADA EM UMA COLABORAÇÃO DE ALGORITMO GENÉTICO E VNS PARA A MINIMIZAÇÃO DE MAKESPAN EM PROBLEMAS DE PROGRAMAÇÃO REATIVA DA PRODUÇÃO

CARLOS CESAR MANSUR TUMA

Tese apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de São Carlos, como parte dos requisitos para a obtenção do título de Doutor em Ciência da Computação, área de concentração: Inteligência Artificial.

Orientador: Prof. Dr. Orides Morandin Junior

São Carlos - SP

Março/2015

Ficha catalográfica elaborada pelo DePT da Biblioteca Comunitária UFSCar
Processamento Técnico
com os dados fornecidos pelo(a) autor(a)

Tuma, Carlos Cesar Mansur
T925e Uma estrutura de vizinhança baseada em árvore de cobertura aplicada em uma colaboração de algoritmo genético e VNS para a minimização de makespan em problemas de programação reativa da produção / Carlos Cesar Mansur Tuma. -- São Carlos : UFSCar, 2015.
175 p.

Tese (Doutorado) -- Universidade Federal de São Carlos, 2015.

1. Árvore de cobertura. 2. Estrutura de vizinhança. 3. Programação reativa da produção. 4. Colaboração global-local. 5. Makespan. I. Título.

UNIVERSIDADE FEDERAL DE SÃO CARLOS

Centro de Ciências Exatas e de Tecnologia
Programa de Pós-Graduação em Ciência da Computação

Folha de Aprovação

Assinaturas dos membros da comissão examinadora que avaliou e aprovou a Defesa de Tese de Doutorado do candidato Carlos Cesar Mansur Tuma, realizada em 31/03/2015:

Prof. Dr. Orides Morandin Junior
UFSCar

Profa. Dra. Roseli Aparecida Francelin Romero
USP

Prof. Dr. Ricardo Augusto Souza Fernandes
UFSCar

Prof. Dr. José Reinaldo Silva
USP

Prof. Dr. Alex Sandro Roschildt Pinto
UFSC

AGRADECIMENTO

À minha esposa Célia e às minhas filhas Ana Beatriz e Marília pelo amor e compreensão.

Aos meus padrinhos Glaura e Aurélio (*in memorian*) pelo apoio e incentivo.

Ao Criador por propor este desafio.

Ao meu orientador, Prof. Dr. Orides Morandin Junior pelas ideias, incentivo e dedicação, além das conversas interessantes.

A todos os professores e funcionários de Departamento de Computação pelas gentilezas e incentivo.

Aos colegas e amigos da pós-graduação pelas trocas de ideias e apoio, em especial ao amigo Vinicius Fernandes Caridá.

À UFSCar a quem dedico este trabalho.

À CAPES pelo apoio financeiro.

RESUMO

A geração de Programação Reativa da Produção (PRP), com o objetivo de minimizar o *makespan*, é uma atividade importante na indústria manufatureira, tendo em vista os numerosos artigos que abordam esta pesquisa na atualidade. Dentre estas pesquisas, destaca-se o uso de hibridização ou colaboração de busca global com busca local, notadamente de Algoritmo Genético (AG) com *Variable Neighborhood Search (VNS)*. Porém, nota-se que as estruturas de vizinhança utilizadas não são correlatas à função de minimização de *makespan* ou, quando o são, são de difícil obtenção. Com o intuito de cobrir tal tópico, esta tese propõe a hipótese de que uma estrutura de vizinhança fortemente correlata ao objetivo de minimização de *makespan* em problemas de PRP, baseando-se em árvore de cobertura e aplicada em uma colaboração de algoritmo genético e VNS, obtém resultados melhores aos obtidos por outros trabalhos, que fazem uso de outras estruturas de vizinhança ou que não utilizam a busca local. A proposta é a construção de um método de colaboração entre AG e VNS usando uma estrutura de vizinhança baseada no mapeamento da solução, em tempo de busca local, na árvore de cobertura associada ao problema, atuando com os operadores *insert*, *swap* e *2-opt*. O planejamento dos experimentos para validação contempla a execução e comparação de quatro variantes de solução de problemas de Programação Reativa da Produção em três cenários de *job shop* de diversas dimensões. Cada par de comparações tem seu tamanho amostral calculado e é examinado com o teste de hipótese adequado. As quatro variantes comparadas são: Algoritmo Genético e três colaborações entre Algoritmo Genético e *Variable Neighborhood Search* (VNS) usando a estrutura de vizinhança proposta e outras duas estruturas de vizinhança (Caminho Crítico e Representação Natural) encontradas na revisão da literatura. Os cenários vem da base Taillard. Os testes corroboram a hipótese com 95% de confiança na comparação com outros trabalhos e a principal contribuição desta tese é a criação de um método eficiente para minimização de *makespan* em PRP.

Palavras-chave: Árvore de cobertura. Estrutura de vizinhança. Programação Reativa da Produção, colaboração global-local, makespan.

ABSTRACT

The generation of Reactive Production Scheduling (PRP) in order to minimize the *makespan* is an important activity in the manufacturing industry, in view of the numerous articles reflecting this search today. Among these studies highlight the global search use in hybridization or collaboration with local search, especially of Genetic Algorithm (GA) with Variable Neighborhood Search (VNS). But see that the neighborhood structures used are not related to the goal of *makespan* minimization or when they are, are difficult to obtain. In order to cover this topic, this thesis proposes the hypothesis that a strongly correlated neighborhood structure with objective of *makespan* minimization in PRP problems, based on spanning tree, and applied on a collaboration among a genetic algorithm with VNS, perform better or equal to those obtained by other studies using other neighborhood structures or without the use of local search. The purpose was to construct a collaboration of GA and VNS using a neighborhood structure based on the mapping of the solution in the spanning tree associated with the problem, in the local search time, and operating with the insert, swap and 2-opt operators. The planning of experiments for validation contemplated since the implementation and comparison of four variants of reactive production scheduling in three job shop scenarios of different sizes. Each pair of comparisons had its calculated sample size and has been tested with the appropriate hypothesis test. The four variants were compared: Genetic Algorithm only and three collaborations of GA with VNS using the neighborhood structure proposal and two other neighborhood structures (Critical Path and Natural Representation) found in the literature review. The scenarios came from Taillard base. The tests corroborate the hypothesis, with 95% confidence, compared to other works and the main contribution of this thesis is to create an efficient method for minimizing makespan in PRP.

Keywords: neighborhood structure, reactive scheduling, spanning tree, makespan, global-local collaboration.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

AC	Árvore de Cobertura
ACO	Otimização por Colônia de Formigas (do inglês: *Ant Colony Optimization*)
AG	Algoritmo Genético
AGV	Veículo Guiado Automaticamente (do inglês: *Automated Guided Vehicle*)
BT	Busca Tabu
CC	Caminho Crítico
FMS	Sistema Flexível de Manufatura (do inglês: *Flexible Manufacturing System*)
HA	Hipótese Alternativa
HN	Hipótese Nula
PO	Programação das Operações
PP	Programação da Produção
PRP	Programação Reativa da Produção
PSO	Otimização por enxame de Partículas (do inglês: *Particle Swarm Optimization*)
RN	Representação Natural
RP	Rede de Petri
SA	Recozimento Simulado (do inglês: *Simulated Annealing*)
UT	Unidade de Tempo
VNS	Busca em Vizinhança Variável (do inglês: *Variable Neigborhood Search*)

SUMÁRIO

Capítulo 1

INTRODUÇÃO

1.1 Contextualização

No cenário das indústrias manufatureiras, a Programação da Produção tem sua relevância acadêmica reconhecida quando consideram-se os numerosos artigos que abordam esta área na atualidade.

Tanto a busca de Programação da Produção (PP), em tempo de planejamento, como a busca de Programação Reativa da Produção (PRP), em tempo de execução, são importantes. Programação no contexto desta tese significa o conjunto de operações da produção, cada operação indicando exatamente que fase de uma tarefa será executada, em qual máquina, e os tempos de início e fim. Programação em tempo de planejamento é quando esta programação é feita com a possibilidade de maior intervenção humana, possivelmente sendo comparada com outras programações e sendo selecionada a que melhor atende aos critérios da fábrica.

Nesta tese, o foco é a PRP, a qual dispõe de um tempo extremamente mais limitado que a PP para ser calculado.

Em PRP, deve-se recalcular a programação devido a problemas como: demanda de matéria-prima, falta de operador, quebra de máquinas e diversos outros fatores. A solução deve ocorrer em um tempo limitado, dentro do horizonte de tempo da programação.

Embora o conceito clássico de PRP seja o indicado no parágrafo anterior, para efeito de validação desta tese, a qual logra a criação de uma programação em tempo limitado, considera-se duas possíveis realidades:

- Pensando-se a tese aplicada em um sistema computacional que contemple PP e PRP: Ao se iniciar a execução de uma PP, percebe-se um evento e decide-se partir para o cálculo da PRP, sendo então que os dados para a PRP são exatamente os mesmos da PP, salvo as alterações geradas pelo evento.
- Pensando-se a tese como um método a ser validado em sua qualidade quanto à limitação de tempo: considera-se que a PP estava em execução, ocorreu um evento, decidiu-se calcular a PRP, e para este cálculo foram passados dados vindos de *benchmarks* para permitir comparações futuras.

Existem vários objetivos possíveis (*makespan, due date, lateness*, etc.) além de suas possíveis combinações. Neste trabalho foca-se no *makespan*, ou tempo necessário para terminar toda uma produção.

Várias técnicas têm sido usadas nas pesquisas buscando otimizar o *makespan*, entre elas encontram-se: Programação Inteira, Filas, Cadeias de Markov, busca global e busca local (heurísticas e meta-heurísticas), hibridizações ou colaborações de busca global com busca local.

Uma das técnicas mais relevantes é o uso de Algoritmo Genético nesta busca. Assim, o uso de Algoritmo Genético em uma hibridização ou colaboração com alguma busca local tornou-se um ramo da pesquisa bastante proficiente.

No contexto desta tese, a hibridização de AG com busca local é definida como sendo a substituição de alguma operação do AG pela busca local, como por exemplo, o cruzamento ou mutação. Já a colaboração de AG com busca local diz respeito a quando, após alguma operação do AG, toda a população é passada para ser tratada pela busca local, mesmo que parcialmente.

Um fator importante na hibridização ou colaboração da busca global com a busca local é que se deve definir claramente qual será a estrutura de vizinhança das soluções, para que se possa aplicar adequadamente as operações de vizinhança nesta estrutura.

Neste trabalho, para evitar a ambiguidade do termo "Programação da Produção (PP)", ora sendo entendido como o problema de planejamento da produção e ora sendo a própria programação obtida, passa-se a considerar o termo "programação das operações" para indicar a sequência de operações incluindo os tempos de início e fim de cada operação.

Outra expressão que poderia gerar dúvidas é "correlação da estrutura de vizinhança com o *makespan*" sendo que a estrutura de vizinhança é uma lista de tarefas e *makespan* é uma medida de tempo. Na presente pesquisa, esta expressão é entendida como sendo a correlação entre as sequências dos finais das operações de cada tarefa da estrutura de vizinhança e a sequência de finais de operações da programação das operações.

Porém, nas pesquisas levantadas na revisão de literatura acerca do assunto, detectou-se que as estruturas de vizinhança utilizadas não possuíam correlações com a programação das operações ou simplificavam demasiadamente esta estrutura, em ambos os casos não fazendo uso desta estrutura de forma eficiente.

Com o objetivo de produzir uma estrutura de vizinhança realmente correlata à programação das operações e desenvolver uma técnica que faça uso inteligente de tal estrutura, propõe-se a tese: Será que uma estrutura de vizinhança mais correlata à Programação das Operações permitirá a criação de uma técnica que obtenha valores de *makespan*, dentro de prazo de processamento limitado, melhores do que os obtidos pelos métodos e estruturas pesquisados?

A proposta deste trabalho é criar e validar um sistema computacional que implemente a colaboração entre AG e VNS, usando na busca local uma estrutura de vizinhança altamente correlata à programação das operações baseada em árvore de cobertura, visando minimizar o *makespan* em problemas de Programação Reativa da Produção.

1.2 Motivação

Cada vez se torna mais necessária a tomada de decisões adequadas sobre projeto, planejamento e operação em sistemas que exigem alocação eficiente de recursos escassos. Essa é a área em que atua a Pesquisa Operacional. "*Pesquisa Operacional é a aplicação de métodos científicos a problemas complexos para auxiliar no processo de tomada das decisões, tais como projetar, planejar e operar sistemas em situações que requerem alocações eficientes de recursos escassos*" (ARENALES et al., 2007, p. IX).

Entre estes problemas: problema da mochila, problemas de mistura, programação da produção, corte, empacotamento, roteamento de veículos, caixeiro viajante entre outros (TAHA, 2008).

Vários métodos científicos e abordagens de otimização são utilizados na busca de suas soluções: programação linear, programação inteira, heurísticas e meta-heurísticas, entre outras (PINEDO, 2005).

Problemas como programação da produção, roteamento de veículos e outros são conhecidos como NP-Completo (ARENALES et al., 2007), o que significa que são intratáveis (não tem como se chegar na conclusão em tempo de processamento) com o uso de modelos matemáticos (Programação Linear e Programação Inteira), sendo, nesses casos, o uso de heurísticas e meta-heurísticas uma solução viável (ARENALES et al., 2007; PINEDO, 2005).

Entre as meta-heurísticas atualmente em uso pode-se citar: Recozimento Simulado (SA – *Simulated Annealing*) (ZHANG et al., 2011; MIRSANEI et al., 2011), Algoritmo Genético (AG – *Genetic Algorithm*) (JARBONI; EDDALY; SIARRY, 2011; SONG e XU, 2010; ZOBOLAS; TARANTILIS; IOANNOU, 2009; TAVARES; CORREIA, 1999), Otimização por Colônia de Formigas (ACO - *Ant Colony Optimization*) (ZHANG; ZHANG; LI, 2010; ESWARAMURTHY; TAMILARASI, 2009), Otimização por Enxame de Partículas (PSO – *Particle Swarm Optimization*) (ZHANG et al., 2011; SUN et al., 2010), Busca Tabu (BT - *Tabu Search*) (LI et al., 2011; ESWARAMURTHY; TAMILARASI, 2009; TAVARES; CORREIA, 1999), Busca em Vizinhança Variável (VNS – *Variable Neighborhood Search*) (JARBONI; EDDALY; SIARRY, 2011; SUN et al., 2010; ZOBOLAS; TARANTILIS; IOANNOU, 2009), entre outros.

Durante a execução de uma programação das operações criada para resolver um problema de Programação da Produção (PP), sempre existe a possibilidade de ser necessário mudar esta programação por questões de alteração de pedido, falta de material, parada de funcionamento de alguma máquina, falta de operador, entre outros. Nesses casos, o objetivo da produção em progresso fica comprometido se uma nova programação não for calculada em tempo hábil. Em tais situações, é necessária a reprogramação da produção, com o agravante da limitação no tempo para seu processamento. A este problema de reprogramação chama-se Programação Reativa da Produção (PRP).

Um dos principais objetivos a ser otimizado em problemas de Programação da Produção e Programação Reativa da Produção é o *makespan* (ARENALES et al., 2007; PINEDO, 2005) e é comum a hibridização ou colaboração das meta-heurísticas de busca global com métodos de busca local, como os seguintes: Algoritmo Genético com VNS (JARBONI; EDDALY; SIARRY, 2011; ZOBOLAS; TARANTILIS; IOANNOU, 2009), PSO com VNS (SUN et al., 2010) e PSO com Recozimento Simulado (ZHANG et al., 2011).

Nos métodos de busca local visita-se a vizinhança de uma solução. Faz-se necessário ter a estrutura de tal vizinhança definida. Várias estruturas estão sendo usadas ultimamente, entre elas: Representação Natural (ZOBOLAS; TARANTILIS; IOANNOU, 2009; SUN et al., 2010; JARBONI; EDDALY; SIARRY, 2011), Caminho Crítico (ESWARAMURTHY; TAMILARASI, 2009; WANG; ZHANG, 2011), Bloco Crítico (HMIDA et al., 2010) e Bloco Crítico Público (LI et al., 2011).

No processo de adequação de um método a um problema específico, é comum o uso de ferramentas de modelagem, as quais permitem definir melhor as estruturas e estratégias de busca de uma boa solução. Vários métodos são utilizados na modelagem de problemas de PP e PRP, sendo que Cadeias de Markov (ARENALES et al., 2007), Sistemas de Filas (ARENALES et al., 2007), Grafos Direcionados (PINEDO, 2005) e Redes de Petri (MURATA, 1989) estão entre os mais usados.

O uso da Rede de Petri como ferramenta para representar problemas de PP e PRP tem sido utilizado em diversas pesquisas (KIM; SUZUKI; NARIKIYO, 2007; JAIN; JAIN; SINGH, 2006; MEJIA; ODREY, 2005; MORO; YU; KELLEHER, 2002), inclusive no Laboratório Tear (MORANDIN et al., 2011; MORANDIN; KATO; TUMA,, 2010; MORANDIN et al., 2007a; MORANDIN et al., 2007b).

A árvore de cobertura obtida a partir da Rede de Petri associada a uma Programação das Operações descreve todas as alternativas de sequências de uso de máquinas. Isso significa que a árvore indica exatamente qual a programação das operações a ser executada em uma produção, sendo altamente correlata ao *makespan* (MURATA, 1989).

Embora a urgência de se recalcular uma programação das operações em execução, em geral, seja tratada no dia a dia pelo responsável pela produção com base em seu conhecimento empírico, uma solução com melhor qualidade poderia ser obtida. Vários métodos propõem-se a melhorar esta qualidade, tanto na área acadêmica como no campo das indústrias.

Neste ponto é que se encontra a motivação desta tese. Criar uma estrutura de vizinhança correlata ao objetivo de minimizar o *makespan* em problemas de PRP aplicada a uma colaboração global-local tendo tempo de processamento computacional como limitação.

1.3 Justificativa

Existem vários métodos e sistemas que se propõem a resolver o problema da PP, inclusive nos casos em que há a necessidade de urgência no tempo de recálculo (PRP).

Porém, na revisão de literatura, ficou claro que a maioria dos métodos de busca local atuais não usam uma estrutura de vizinhança correlacionada ao *makespan*. Os métodos que possuem uma estrutura de vizinhança mais correlata ao *makespan* pecam pela complexidade de seus cálculos e/ou pelo excesso de simplificação desta estrutura, se afastando demasiadamente da realidade.

Nos artigos pesquisados, a definição da estrutura da vizinhança ora era tratada como a própria programação (MIRSANEI et al., 2011; YAZDANI; AMIRI; ZANDIEH, 2010; HAN; DUAN; ZHANG, 2010), ora como uma simplificação da mesma (WANG et al., 2012; LI et al., 2011; ZHANG; ZHANG; LI, 2010) e o modo de efetuar a busca na vizinhança se dava pela troca de duas operações da programação (RIBAS; COMPANYS; TORT-MARTORELL, 2011; HMIDA et al., 2010; ZHENG; YAMASHIRO, 2010) ou pela modificação da posição na programação em que uma operação iria ocorrer (PAN et al., 2011a; SELS; CRAEYMEERSCH; VANHOUCKE, 2011; ZHANG; ZHANG; LIANG, 2010), ou, ainda, pela inversão de uma sequência de operações dentro da programação (TRAN; NG, 2011; GILAK; RASHIDI, 2009; ZOBOLAS; TARANTILIS; IOANNOU, 2009).

Nota-se que não se evidencia nesses tratamentos de vizinhança uma correlação com a variável implicada (*makespan* da programação das operações). Uma possível vizinhança mais correlacionada ao problema deve levar a melhores resultados, pois parece provável que melhorando a definição da vizinhança pode-se aumentar a eficiência de outros métodos locais.

1.4 Objetivos

O objetivo geral desta tese é definir uma estrutura de vizinhança baseada em árvore de cobertura e uma estratégia de uso de suas propriedades, almejando a minimização de

makespan para problemas de PRP. Esta estrutura será aplicada em uma colaboração de busca global com busca local e pretende-se obter resultados melhores do que os obtidos por trabalhos usando outras estruturas de vizinhança. A tese é uma contribuição na direção da solução do problema de PRP.

Já os objetivos específicos são:

- Definir o método de busca global AG e o método de busca local VNS como métodos a serem postos em colaboração na busca de menor *makespan*, com tempo de processamento limitado;
- Definir todos os parâmetros a serem usados na implementação da colaboração;
- Definir e implementar a estrutura de vizinhança baseada em árvore de cobertura;
- Selecionar os cenários do problema de programação de produção j*ob shop* como contexto a ser usado na validação e definir a base de cenários de Taillard como fonte dos cenários usados;
- Escolher as estruturas de vizinhança Representação Natural e Caminho Crítico como estruturas de comparação na fase de validação e implementá-las.

1.5 Delimitações do trabalho

O problema de Programação Reativa da Produção inclui diversas tarefas que podem ser realizadas por meio de roteiros diferentes entre si. O que se busca é uma PO que corresponda ao menor *makespan* possível em um limite de tempo de processamento considerado curto.

São consideradas algumas delimitações aos cenários propostos, relatadas a seguir:

- Cada máquina opera apenas uma tarefa por vez e cada tarefa só pode ser operada por uma máquina por vez;
- Os tempos de operação são determinados e previamente conhecidos;
- Os tempos de *setup* e transportes estão inclusos nos tempos de operação;
- Inicia-se o roteiro de uma tarefa transportando a matéria-prima do setor de carga para o *buffer* de entrada da primeira máquina em seu roteiro;
- Termina-se o roteiro de uma tarefa transportando-se a tarefa finalizada do *buffer* de saída da última máquina a operá-la para o setor de Descarga;
- Toda máquina possui *buffer* de entrada e *buffer* de saída de tamanho grande o

suficiente para suportar qualquer demanda;

- Máquinas não param e nem dão defeito.

A princípio não serão tratados:

- Tempos de *setup;*
- Uso de AGVs (*Automated Guided Vehicles*);
- Tarefas com componentes;
- Tarefas que são tratadas mais de uma vez na mesma máquina;
- Lotes de tamanhos diferentes do padrão;
- Lote de saída de uma etapa com número diferente do lote de entrada na fase seguinte;
- *Buffer* de entrada e saída limitado.

1.6 Método de pesquisa e desenvolvimento

Esta tese segue o método de pesquisa Hipotético-Dedutivo (POPPER, 2007; LAKATOS; MARCONI, 2003) o qual propõe que dada uma hipótese busca-se falseá-la. Se os resultados não a falsearem considera-se a hipótese corroborada, caso contrário a hipótese é reformulada e novos testes são feitos, reiniciando o ciclo. A hipótese desta tese é que uma estrutura de vizinhança mais correlata à variável do problema (minimização de *makespan*) resulte em uma solução melhor que as soluções encontradas usando outras estruturas de vizinhança, sendo que a corroboração da hipótese ocorre pelo não falseamento de que os resultados obtidos pelo uso da estrutura de vizinhança proposta são melhores do que os resultados obtidos pelo uso de outras estruturas de vizinhança.

Segundo Popper (2007):

- A hipótese não é defendida para provar que está certa, mas procura-se demonstrar que é falsa. Caso não se consiga falseá-la, diz-se que "comprovou sua qualidade" ou foi "corroborada". As hipóteses não são verificáveis, mas podem ser corroboradas.
- Os métodos estatísticos são essencialmente hipotético-dedutivos.

Definiram-se as seguintes etapas a serem seguidas:

- Levantamento bibliográfico:
  - Levantar-se artigos que contemplem busca na vizinhança em problemas de

PRP e PP objetivando melhor *makespan* com vistas a verificar a relevância e atualidade da pesquisa e, principalmente, garantir o ineditismo da proposta. Esta pesquisa busca encontrar métodos e estruturas usadas nas pesquisas atuais em problemas correlatos ao da tese;

- Deste levantamento se extrai informações sobre os métodos de busca global e local, estruturas de cromossomo e vizinhança e operadores de vizinhança. Essas informações guiam a definição da proposta e a forma como se dá a sua validação.

- Modelagem e Implementação da proposta:
  - Nesta etapa, definem-se os métodos de busca, operadores de vizinhança e a estrutura proposta;
  - Define-se a colaboração do método global (algoritmo genético) com o método de busca local (VNS), utilizando os operadores de vizinhança *Insert*, *Swap* e *2*-op;
  - Define-se a estrutura de cromossomo como sendo uma sequência de sugestões das tarefas para a programação das operações (PO), conhecida como cromossomo baseado em operações (CHENG; GEN; TSUJIMURA, 1996);
  - Define-se a estrutura de vizinhança como sendo a lista sequencial das tarefas das operações que ocorrem na PO, agrupando as tarefas com mesmo início de operação em tarefas virtuais;
  - Define-se usar as equações de estados associadas à árvore de cobertura como estratégia de aproveitamento de parte da simulação feita em tempo de busca local;
  - Como ferramenta de Programação é usado o ambiente de programação Matlab;
  - Um sistema de computador é montado para permitir a verificação da viabilidade da proposta assim como de sua comparação com outras estruturas.

- Validação:
  - Nesta etapa, o sistema construído é modificado com o objetivo de se poder avaliar a hipótese quando comparada com outras estruturas de vizinhança. As

modificações permitem:

- A execução do sistema usando somente o AG, sem intervenção da busca local;
- A execução da colaboração com o uso de duas outras estruturas de vizinhança no lugar da estrutura de vizinhança proposta;

- Para tal comparação, os cenários descrevem o problema de j*ob shop* e são importados da base Taillard (TAILLARD, 1993; TAILLARD, 2015);
- Um planejamento dos experimentos é feito e as diversas configurações do sistema são executadas e seus resultados são analisados com o auxilio de ferramentas de estatística. Esta análise permite corroborar a hipótese levantada de que uma estrutura de vizinhança mais correlata ao *makespan* obtém resultados melhores no contexto testado.

1.7 Organização do trabalho

Como este trabalho se pauta na metodologia hipotético-dedutiva, inicialmente faz-se uma revisão sistemática para analisar o estado da arte da pesquisa em conceitos de vizinhança em métodos de busca local que minimizam o *makespan* em problemas de Programação da Produção e Programação Reativa da Produção (Capítulo 2).

Para deixar claros os conceitos levantados no Capítulo 2, logo no início deste mesmo capítulo, faz-se uma breve apresentação desses conceitos desde a definição do problema até a descrição dos algoritmos utilizados. O Apêndice B define os métodos estatísticos de comparação a serem aplicados.

Tecendo então a hipótese de que uma vizinhança mais fortemente relacionada ao *makespan* da programação das operações deve permitir gerar resultados melhores do que os resultados dos trabalhos atuais, esta proposta de vizinhança é apresentada no Capítulo 3 assim como sua modelagem em Rede de Petri e exemplos para facilitar o entendimento.

Apresenta-se no Capítulo 4 como se pretende falsear esta hipótese e isto se dá pela comparação dos resultados obtidos pela aplicação das vizinhanças e operadores definidos no Capítulo 2 e os resultados obtidos pela vizinhança e operadores desta proposta. Como contexto são usados cenários que devem estar próximos à realidade de sistemas de produção e estes são oriundos das bases indicadas no Capítulo 4. Estes cenários estão tabelados no

Apêndice A. As comparações são validadas com métodos estatísticos e corrobora-se a hipótese por não conseguir falseá-la. Todas as tabelas de dados, resultados e análises estão no Apêndice C.

O Capítulo 5, nomeado Conclusão, fecha o trabalho analisando os resultados e indicando possíveis trabalhos futuros.

Capítulo 2

ESTADO DA ARTE

2.1 Considerações iniciais

Visando produzir uma estrutura de vizinhança realmente correlata à programação obtida e desenvolver uma técnica que faça uso inteligente de tal estrutura, propõe-se a tese: Será que uma estrutura de vizinhança mais correlata à programação obtida permitirá a criação de uma técnica que obtenha valores de *makespan*, dentro de um prazo de processamento limitado, melhor do que os obtidos por outros métodos e estruturas pesquisados?

Neste capítulo, serão explicados conceitos importantes para o entendimento da revisão e da proposta, tais como as representações do problema de PRP, do espaço de soluções e das estruturas de vizinhança, assim como o funcionamento dos operadores de vizinhança e como se costuma buscar a solução de PRP. Após estas conceituações, será apresentada a revisão de literatura e sua análise, mostrando a influência que têm na definição da proposta desta tese.

2.2 Fundamentação teórica

Para que se possa entender e propor uma solução para um problema é necessário equacioná-lo. Os primeiros passos para isto estão na definição de tal problema, suas delimitações e a forma como representá-lo, assim como seu espaço de soluções. A partir deste ponto pode-se propor um método de busca de soluções e, caso se use uma busca no entorno de uma solução, deve-se definir a estrutura de vizinhança e as formas de passar (operadores) de uma solução desta vizinhança para outra.

Esta seção apresenta estes conceitos: definição do problema de PRP, suas representações e de seus espaços de soluções, as estruturas de vizinhança e operadores

comumente usados. Além de apresentar como se busca suas soluções e apresentam-se os métodos Algoritmo Genético e VNS.

2.2.1 **Descrição do Problema de PRP e suas Soluções**

PRP começa de uma condição de fábrica, quando faz-se necessário que a programação vigente seja atualizada devido à algum evento que a torna não mais interessante. Tal evento pode ser desde a falta de matéria prima a falta de operador, quebra de máquina ou mudança no pedido atual, ou outras possibilidades (ARENALES et al., 2007).

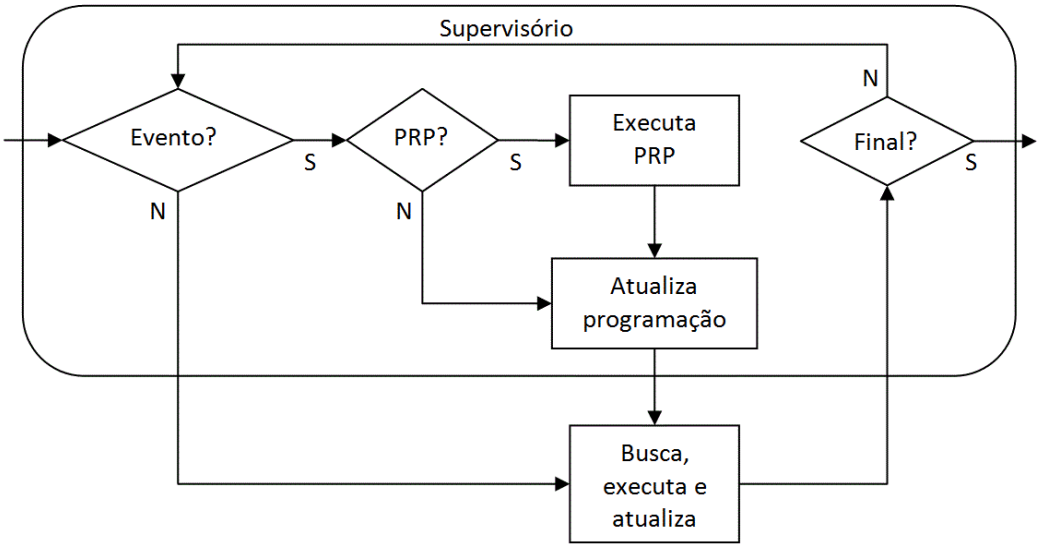

Figura 2.1 Diagrama de um sistema PP/PRP, contemplando o supervisório e o chão de fábrica.

A Figura 2.1 mostra um diagrama representando uma descrição possível de um sistema supervisório, controlando a produção (executando a programação recebida) e trabalhando sobre possíveis eventos, decidindo ou não fazer uma reprogramação (PRP).

O processo começa verificando se ocorreu algum evento proibitivo. Se tiver ocorrido verifica-se se é necessário fazer uma reprogramação e, caso se justifique, passa ao programa de PRP todos os dados referentes a programação em curso (lotes, cenários e situação das marcações e transições da Rede de Petri associada) e o tempo para que tal reprogramação seja calculada.

Uma vez calculada a nova programação, a programação em curso é substituída, com os devidos ajustes (operações já feitas, e executando).

Caso a reprogramação não seja necessária, apenas ajustes são feitos na programação (produtos que entraram ou saíram, troca de máquinas, etc.).

Resolvido o evento e possível reprogramação, a atividade passa para o chão de fábrica, buscando a operação a ser iniciada ou finalizada, executando-as e atualizando a programação atual.

Embora o conceito de PRP diga que ela ocorre em tempo de execução de uma PP, que foi calculada anteriormente, e por conta de algum evento proibitivo da execução normal dessa PP aciona o seu recálculo gerando uma PRP em tempo curto. Em um sistema de controle de PP/PRP ao ocorrer um evento, este deveria ser ponderado e, caso se julgue necessário gerar nova programação (PRP), a situação das marcações e árvore de cobertura da Rede de Petri são passadas ao recálculo, considerando a situação ao tempo final do recálculo. Caso ocorra um evento no tempo x, e caso o recálculo seja estimado em um tempo *y*, são passados como parâmetros da PRP: a situação das marcações e Rede de Petri no tempo $x + y$.

Porém o objetivo desta tese é verificar a eficiência de um método de busca baseado em árvore de cobertura e, para efeito de validação e comparação, se considera que a PP estava em execução, ocorreu um evento, este evento disparou um recálculo de PP, e os parâmetros de execução da PRP são os cenários usados como entrada na validação. Considera-se também que o tempo *y* é estimado em 5 minutos. Caso se pretendesse manter um supervisório da Rede de Petri gerada, o uso de árvore de cobertura já não seria adequado e sim a árvore de alcançabilidade. Mas no escopo desta tese, árvore de cobertura é adequada.

O problema tratado nesta proposta é de Programação Reativa da Produção, que é assim definido: como gerar todas as operações de um conjunto de tarefas, com o fim de minimizar algum objetivo, em um tempo de processamento relativamente curto (PINEDO, 2005).

Estas operações devem informar seus tempos de início e final e, com este conjunto, pode-se descrever exatamente todo o processo da produção, sabendo em que intervalo de tempo determinada operação vai executar.

Entre as restrições normalmente encontradas estão:

- Cada máquina só executa uma tarefa por vez e cada tarefa só pode ser executada em uma máquina por vez;

- Não se admite preempção;
- Os tempos de configuração e transporte são considerados embutidos nos tempos de operação.

Ambientes de produção

Entre os ambientes de produção pode-se citar: *job shop*, *flow shop* e *open shop*.

"Um *job shop* clássico é um ambiente de produção com `n` tarefas e `m` máquinas, em que cada tarefa é processada nas `m` máquinas, de acordo com um roteiro pré-estabelecido" (ARENALES et al., 2007, pag. 222).

A diferença entre *job shop, flow shop* e *open shop* é: no *job shop* cada tarefa possui um roteiro, no *flow shop* todas as tarefas possuem o mesmo roteiro e no *open shop* não há roteiro.

Cada um destes ambientes pode possuir várias variantes, como *buffers* limitados (em que se limita a capacidade dos *buffers*) ou *no-wait* (em que não se admite intervalo de tempo de espera entre uma operação e outra da mesma tarefa), entre outros.

Cenários e Desempenho

Os dados de entrada dos problemas são representados por um conjunto de dados chamados de cenários e podem ser entendidos como a configuração de um problema de programação, indicando as tarefas a serem programadas com seus tempos de execução e roteiros. Tais dados podem incluir múltiplos roteiros para cada tarefa, assim como tempos de configuração e transporte, mas estes atributos não interessam ao escopo desta tese.

A avaliação de um método de busca depende de cenários nos quais possam ser geradas soluções. A validação depende das comparações de resultados da proposta com os obtidos de outros trabalhos.

Estes cenários podem ser gerados aleatoriamente, vir de especificações constantes na literatura ou mesmo ser encontrados em sítios na *Internet*, já prontos para o uso. Tanto nas especificações na literatura como nos sítios na *Internet*, muitas vezes, encontram-se informações valiosas sobre o *makespan* conhecido para seus problemas.

Entre os artigos que fornecem instruções para a criação de cenários estão os do Prof. Dr. Eric Taillard da *Haute École d'Ingénierie et de Gestion du Canton de Vaud* (APHEIG-VD), frequentemente referenciados na literatura (TAILLARD, 1993). Este mesmo pesquisador e seu grupo de pesquisa mantêm um sítio na *Internet* (TAILLARD, 2015) com vários cenários prontos, também frequentemente referenciados.

Entre as medidas de desempenho pode-se citar: *makespan*, atraso máximo, *lateness*, entre outros. No escopo desta tese só o *makespan* foi considerado.

Makespan é o tempo necessário para que todas as tarefas sejam completadas, e o objetivo de mínimo *makespan* ou mínimo tempo de completude é aquele que se busca o menor tempo possível para um determinado conjunto de tarefas a serem executadas (PINEDO, 2005).

Nos *benchmarks*, considera-se que cada tarefa passa por todas as máquinas apenas uma vez e o lote de tarefas alocado costumeiramente é de uma unidade por tarefa, chamado de lote padrão.

Nas próximas seções são apresentadas as representações do problema de PRP, de seus espaços de soluções, das estruturas de vizinhança mais utilizadas nestas representações e da forma como se busca obter solução para este problema.

2.2.1.1 Representação do problema de PRP

Comumente os artigos pesquisados não informam a forma como representaram o problema a ser resolvido, mas alguns modelos são os habituais.

Existem várias formas de representação e entre as formas mais comuns pode-se citar: vetor de permutação, grafos orientados, Redes de Petri, Cadeias de Markov e Filas.

Como as estruturas de vizinhança encontradas na revisão de literatura usam somente as três primeiras representações dadas acima, optou-se por não descrever Cadeias de Markov e Filas. Entre os motivos para tal ausência na literatura talvez estejam: representação gráfica complicada a medida que o número de recursos sendo compartilhados.aumenta; uso de

tempos estocásticos; dificuldade de representar concorrência em Filas; dificuldade de definir vizinhança de operações em Markov.

Os dados que instanciam o problema costumam estar na forma de tabelas contendo os tempos de duração de cada operação e as máquinas que executam estas operações, formando seus roteiros.

Por exemplo:

Considere as tabelas 2.1 e 2.2 que definem um problema de programação de *job shop*. Este exemplo é referenciado em vários momentos neste e no próximo capítulo. A sigla ut significa unidade de tempo.

Tabela 2.1 Roteiros para um conjunto de tarefas do exemplo, indicando as máquinas de cada operação.

	Primeira operação	Segunda operação	Terceira Operação
Tarefa 1	m1	m2	m3
Tarefa 2	m1	m2	m3

Tabela 2.2 Tempos de operação de cada fase de um conjunto de tarefas do exemplo.

	Tempo operação 1	Tempo operação 2	Tempo operação 3
Tarefa 1	30 ut	9 ut	5 ut
Tarefa 2	10 ut	10 ut	20 ut

Espaço de soluções

O espaço de soluções permite ter uma ideia da dimensão do tamanho do problema. Para problemas de *job shop*, em que todos os roteiros têm m operações, o cálculo do número de soluções é dado por

$$ns = (n \times m)!/(m!^n),$$

Onde n é o número de tarefas e m o número de máquinas.

Assim, nota-se que para um cenário de duas tarefas por três máquinas, daria 6!/(3!^2) = 20, em contrapartida, para um cenário de seis tarefas por três máquinas, daria 18!/(3!^6) = 1,37 E+11.

Estrutura da solução

Uma solução indica todas as operações e os momentos em que são executadas na programação. É representada por um conjunto de tuplas de operações, correspondendo a todas as operações desta solução de programação. Estas tuplas são compostas por cinco valores que indicam respectivamente: tarefa, operação, máquina, tempo de início e tempo de fim.

Uma solução genérica é o conjunto de tuplas

$SG = \{(j_k, e_k, r_k, i_k, f_k)\ tal\ que\ \ k = 1, (número\ de\ operações)\}$,

Onde `j` indica o *job*, o `e` a etapa, `r` a máquina , `i` o tempo de início e `f` o tempo final.

Considerando as tabelas 2.1 e 2.2, uma possível programação para este problema é o seguinte conjunto de tuplas:

((1,1,1,0,30), (1,2,2,30,39), (2,1,1,30,40), (1,3,3,39,44), (2,2,2,40,50), (2,3,3,50,70))

Analisando a tupla (1,3,3,39,44), tem-se que:

- O primeiro elemento indica a tarefa desta operação, tarefa 1;

- O segundo elemento indica a operação a ser executada, operação 3;

- O terceiro elemento indica a máquina que vai operar esta tarefa nesta operação, esta informação vem da Tabela 2.1, linha 2 (tarefa 1) coluna 4 (operação 3) e aponta a máquina 3;

- O quarto elemento indica o tempo de início da operação 3, da tarefa 1 na máquina 3 e este valor vem do método de programação, podendo ser o próximo tempo em que a máquina 3 estiver livre. Todas as máquinas iniciam a programação estando livres. No referido caso, o tempo é de 39 unidades de tempo (ut);

- O quinto elemento indica o tempo de término da operação 3 da tarefa 1 na máquina 3, que começou no tempo de 39 ut. Este valor é obtido pela soma do tempo de início com o tempo de processamento da operação 3 da tarefa 1. Este tempo pode ser observado por meio da Tabela 2.2, linha 2 e coluna 4 e é igual a 5. Então o valor de término da operação é 39 + 5 = 44 ut. .

Como este problema de programação é composto por duas tarefas com três operações cada, são necessárias 6 tuplas de operações para definir adequadamente a programação.

Esta mesma programação pode ser interpretada pelo gráfico de Gantt dado na Figura 2.1. Os números dentro das caixas no gráfico indicam as tarefas.

Nota-se que a tarefa 1 tem sua primeira operação acontecendo na máquina 1, tupla (1,1,1, 0, 30); sua segunda operação acontecendo na máquina 2, tupla (1,2,2, 30, 39); e sua última operação acontecendo na máquina 3, tupla (1,3,3,39, 44). Para esta tarefa observa-se que a produção foi iniciada no tempo 0 e foi finalizada no tempo 44.

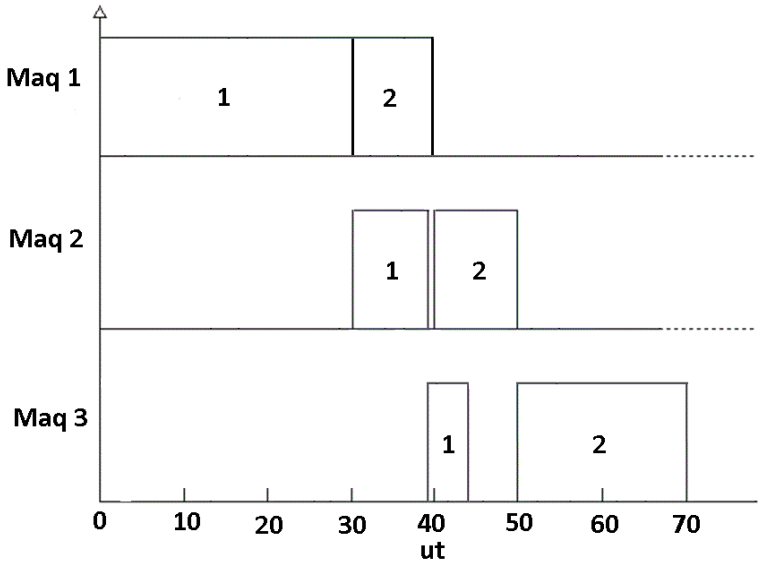

Figura 2.2 Gráfico de Gantt para o exemplo dado.

Na programação apresentada estão duas tarefas. Cada uma delas com três operações, somando um total de 6 tuplas de operações, que estão igualmente divididas entre as máquinas (3 em cada). O *makespan* desta programação é de 70 ut.

Estrutura da Vizinhança

Ao considerar-se uma solução e decidir-se fazer uma busca em seu entorno, deve-se definir qual é esse entorno ou vizinhança. Pode-se considerar que vizinhanças são somente determinados subconjuntos de tarefas da programação, ou mesmo ser a programação completa. A busca na vizinhança ocorre aplicando-se operadores de vizinhança na estrutura de vizinhança definida.

Estruturas de vizinhança são as estruturas que, quando sofrem a operação de algum operador de vizinhança, levam de um vizinho a outro. Devem possuir a propriedade de que um vizinho possa alcançar qualquer outro dentro da estrutura e que, de qualquer um, possa retornar ao vizinho original (AARST; LENSTRA, 2004).

Nas próximas subseções, explicam-se as formas de representação do PRP encontradas na revisão da literatura feita nesta tese. Para cada forma de representação, define-se também a representação do espaço de soluções e da estrutura de vizinhança.

2.2.1.1.1 Vetor de Operações

Uma das formas mais clássicas de representar um problema de PRP é através de um vetor de operações, no qual cada posição indica uma operação atribuída a uma fase de uma das tarefas que compõem o problema.

Um vetor genérico é dado por $(o_1, o_2, \ldots, o_{p-1}, o_p)$, onde o_i indica uma operação referente a alguma fase de determinada tarefa, o vetor tem p operações e p é igual ao número de tarefas multiplicada pelo o número de máquinas.

Como exemplo, a representação do problema apresentado pelas tabelas 2.1 e 2.2 é o seguinte:

$(o_1, o_2, o_3, \ldots, o_5, o_6)$ sendo que cada o_i representa uma das 3 fases de uma tarefa especifica e se o_i e o_j representam a mesma tarefa e $i < j$, então a operação indicada por o_i deve ser precedente a operação o_j.

O problema é representado por uma permutação de repetições, onde cada posição é uma operação. Esta forma de representação deixa bem claro, matematicamente, o tamanho da explosão de complexidade de cada problema. Por exemplo: para um problema com duas tarefas e três máquinas, como o do exemplo anterior, o número de soluções possíveis é da ordem de (2x3)!/(3!^2) = 20.

Cada solução possível é a instanciação de uma das permutações possíveis. Para o exemplo instanciado anteriormente, cuja programação possível foi dada, sua representação natural é: (1,1,1,2,2,2), que é a sugestão de sequencialização das tarefas para a programação.

Deve-se notar que a sugestão e a programação são diferentes, pois a sugestão é uma sequência de tarefas sugeridas enquanto a programação é um conjunto de operações a serem executadas, com tempos de início e fim associados.

Nesta representação, a sequência indicada é uma sugestão de programação, mas a ordem real só será definida durante a simulação.

O espaço de soluções da Vetor de operações - Matriz de permutações

O espaço de soluções é o conjunto de todos os vetores contendo uma solução possível. Pode ser entendido também como uma matriz com número de colunas igual ao número de operações de uma programação e com número de linhas igual ao número de soluções possíveis. Graficamente é de visualização difícil, pois somente exemplos muito pequenos podem ser assim representados para inspeção visual.

Por exemplo, para o problema dado anteriormente, o espaço de soluções como matriz de permutações é o seguinte:

1	1	1	2	2	2
1	1	2	1	2	2
1	1	2	2	1	2
1	1	2	2	2	1
1	2	1	1	2	2
1	2	1	2	1	2
1	2	1	2	2	1
1	2	2	1	1	2
1	2	2	1	2	1
1	2	2	2	1	1
2	1	1	1	2	2
2	1	1	2	1	2
2	1	1	2	2	1
2	1	2	1	1	2
2	1	2	1	2	1
2	1	2	2	1	1
2	2	1	1	1	2
2	2	1	1	2	1
2	2	1	2	1	1
2	2	2	1	1	1

Figura 2.3 Espaço de soluções para o exemplo dado.

A Estrutura de Vizinhança do Vetor de Operações - Representação Natural

Nesta estrutura de vizinhança, considera-se a sequência de sugestão de tarefas geradora da programação da produção inteira como vizinhança possível. Esta é a estrutura de vizinhança mais encontrada [ROSHANAEI et al. , 2009; JARBONI; EDDALY; SIARRY, 2011; ZOBOLAS; TARANTILIS; IOANNOU, 2009; SONG; XU, 2010; SUN et al. , 2010; PAN et al. , 2011b; YAZDANI; AMIRI; ZANDIEH, 2010; ZHENG; YAMASHIRO, 2010; TANG et al. , 2011]

Exemplo:

Tomando a programação dada como exemplo anteriormente, a estrutura da vizinhança é: (1,1,1,2,2,2).

Note-se que a sequência de tarefas é a mesma do exemplo anterior, pois esta forma de representação é considerada natural por ser equivalente à representação da sugestão da solução.

As operações de vizinhança ocorrem nesta sequência de tarefas, trocando posições, retirando e reinserindo uma tarefa em outra posição ou mesmo invertendo parte da sequência. Após uma transformação de uma sequência em outra, as programações são recalculadas, sendo que as novas programações obtidas, provavelmente, serão diferentes.

Nesta estrutura de vizinhança a solução alterada é igual à estrutura de vizinhança alterada. Após a operação na vizinhança basta trocar a solução inicial pela estrutura operada.

Os exemplos dos operadores de vizinhança apresentados mais à frente neste Capítulo usam esta estrutura de vizinhança como padrão.

2.2.1.1.2 Grafos Disjuntivos Orientados com precedência

O problema da Programação Reativa da Produção, em ambientes de *job shop* ou *flow shop,* pode ser representado por um grafo disjuntivo orientado com precedência, onde os nós são as operações e os arcos conjuntivos indicam os tempos de produção e a ordem em que as operações de uma determinada tarefa devem ser executadas, seguindo o roteiro pré-estabelecido, e os arcos disjuntivos indicam as operações que usam as mesmas máquinas. Aos

nós representando as operações, dois nós virtuais são acrescentados, um indicando a operação de início e o outro indicando a operação de término.

Uma programação depende da escolha dos sentidos dos arcos disjuntivos, transformando-os em conjuntivos.

Um grafo disjuntivo é dado por $G = \{V,C,D\}$ onde V é o conjunto dos vértices representando as operações, C é o conjunto dos arcos conjuntivos representando a precedência das operações para cada tarefa e, D é o conjunto dos arcos disjuntivos representando as tarefas que ocorrem na mesma máquina.

Como exemplo, a Figura 2.3 mostra o grafo disjuntivo para o cenário de *job shop* dado no exemplo deste capítulo. É um grafo disjuntivo orientado com precedência composto de vértices que indicam as operações V, cada operação indicando como primeiro elemento a máquina que a executa e como segundo elemento a tarefa da qual faz parte. Os vértices com o segundo elemento igual, mesma tarefa, são ligados sequencialmente por uma aresta contínua, que indica a precedência das operações da tarefa. Dois vértices virtuais, início e término, indicam o início e o final das operações. Os arcos tracejados com duas setas são os arcos disjuntivos, os quais ligam as operações que ocorrem na mesma máquina (adaptado de PINEDO, 2005).

A programação é representada pela soma dos caminhos, com pelo menos um deles indo do nó início até o nó término, transformando os arcos disjuntivos no seu caminho em arcos conjuntivos. Cada caminho deve passar por apenas um nó em V.

O espaço de busca desta forma de representação do problema de programação é o conjunto de todos os conjuntos destes caminhos.

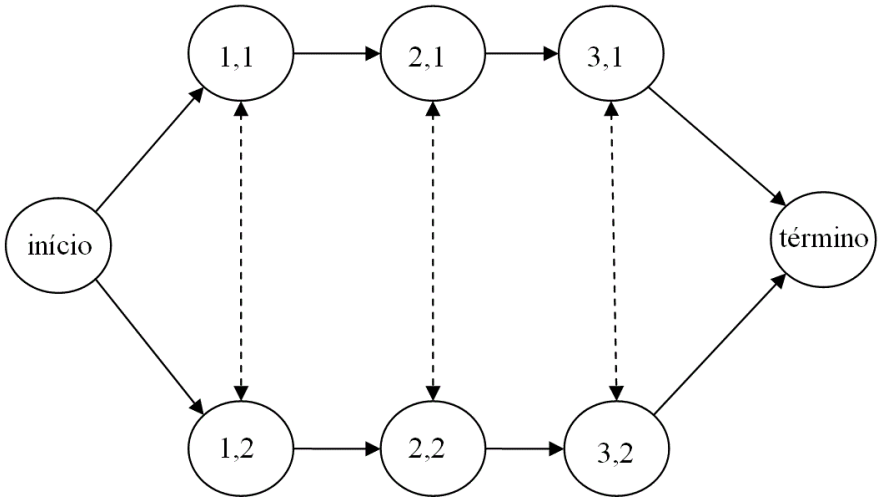

Figura 2.4 Grafo disjuntivo para o exemplo dado (adaptado de Pinedo, 2005).

Estrutura de Vizinhança Caminho Crítico - CC

Caminho Crítico é o caminho mais longo dentro de uma programação correspondendo ao *makespan*. É o trecho de programação iniciando no tempo zero e terminando no tempo igual ao *makespan*, cujas operações são críticas pois se atrasadas aumentam o *makespan* da programação geral (PINEDO, 2005).

A estrutura de vizinhança chamada Caminho Crítico pode ser obtida do grafo orientado disjuntivo com precedência, representando o problema, indicando o maior caminho que leva do início ao término, tendo *makespan* igual ao da programação obtida. Este CC é mostrado na Figura 2.4 em que se pode ver o caminho no grafo, onde se transformou alguns arcos disjuntivos em conjuntivos.

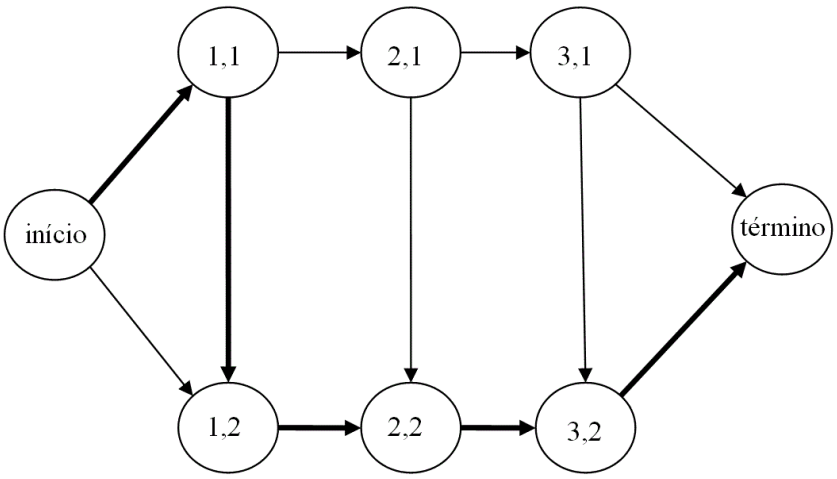

Figura 2.5 Caminho Crítico no grafo disjuntivo para o exemplo dado (adaptado de Pinedo, 2005).

Pode ser calculado pelo algoritmo dado na Figura 2.5. Este algoritmo permite calcular todos os caminhos críticos dentro de uma programação com *makespan* conhecido. Este algoritmo é inspirado no algoritmo apresentado em (ARENALES et al., 2007).

Como exemplo de uso do algoritmo da Figura 2.5 aplicado à programação obtida gerada neste Capítulo pode-se ver o Caminho Crítico obtido na Figura 2.4.. Note que o *makespan* do Caminho Crítico é o mesmo da programação. Abaixo especifica-se mais cada conjunto de operações trabalhado.

Programação: ((1,1,1,0,30), (1,2,2,30,39), (2,1,1,30,40), (1,3,3,39,44), (2,2,2,40,50), (2,3,3,50,70)).

Conjunto de operações do Caminho Crítico: ((1,1,1,0,30), (2,1,1,30,40), (2,2,2,40,50), (2,3,3,50,70)).

Note que o tempo de início da primeira operação é igual ao tempo zero e o tempo de término da última operação é igual ao *makespan*. Além disso, os tempos de término das operações intermediárias são iguais aos tempos de início das operações subsequentes.

Estrutura de vizinhança Caminho Crítico: (1, 2, 2, 2).

A estrutura de vizinhança obtida é a sequência de tarefas das operações do Caminho Crítico.

Para a reinserção da estrutura de vizinhança operada (vizinho), faz-se a substituição das tarefas associadas da programação obtida da solução com a vizinhança obtida.

Exemplo:

Da solução inicial (1,1,1,2,2,2) obteve-se a programação correspondente ((1,1,1,0,30), (1,2,2,30,39), (1,3,3,39,44), (2,1,1,30,40), (2,2,2,40,50), (2,3,3,50,70)) e o conjunto de operações do Caminho Crítico ((1,1,1,0,30), (2,1,1,30,40), (2,2,2,40,50), (2,3,3,50,70)) e a Estrutura de vizinhança Caminho Crítico (1, 2, 2, 2).

Correspondendo as posições do Caminho Crítico com as posições na solução (1/1, 2/4, 2/5, 2/6).

```
Algoritmo de cálculo de Caminho Crítico
Prog -  programação pronta
Mkp – makespan de Prog
Pilha – pilha de operações para busca de caminho crítico
Contadorpilha – quantos níveis ativos na pilha
Operacoespilha – quantas operações por nível da pilha

Buscar em Prog todas as operações que tem final igual ao Mkp
Inserir na Pilha e atualizar controles
AnalisaPilha()

Função AnalisaPilha()
Se Pilha vazia,
        terminar análise
senão
        Obter última operação empilhada
        Se tempo início da última operação empilhada = 0
            Então
                Retornar Caminho Crítico encontrado, resgatando todas
                as operações nos topos da Pilha
                Ajustar Pilha e controles
                AnalisaPilha()
            Senão
                Buscar as operações em Prog que o tempo de final é
                menor ou igual ao tempo de início da operação
                empilhada, mas só pega as que tiverem tempo final
                igual ao máximo
                Inserir na Pilha e atualizar controles
                AnalisaPilha()
```

Figura 2.6 Pseudocódigo da busca dos Caminhos Críticos (adaptado de Arenales et al., 2007).

Supondo que a estrutura fosse alterada para (2, 1, 2, 2), a solução seria alterada nas posições 1 e 2 ficando a solução alterada (2, 1, 1, 1, 2, 2).

2.2.1.1.3 Rede de Petri

A Rede de Petri possui características favoráveis para a modelagem de sistemas que possuem concorrência por recursos, permitindo diversas abordagens, indo desde uma resolução de conflito por regra padrão, decisões nebulosas ou mesmo interferência de operador. Por estes motivos esta modelagem é usada nesta proposta [MURATA, 1989].

Conceitos Básicos

"Uma rede de Petri é um tipo particular de grafo orientado, bipartido, junto com um estado inicial chamado marcação inicial M_0.

A Rede de Petri Lugar-Transição é composta de dois tipos de nós: lugares, representados por um círculo, e transições, representados por barras ou quadrados.

São ligados por arcos unidirecionais que podem ter pesos associados a eles. Esses pesos são inteiros positivos. Assume-se valor 1 quando não indicado.

Marcas são colocadas em lugares e são simbolizadas por bolinhas pretas" [MURATA, 1989].

Marcações são indicadas por $\mathtt{M}$ e são m-vetor, em que $\mathtt{m}$ é o número de lugares na rede. Indica-se por $M(x)$ o número de marcas no lugar $\mathtt{x}$ da marcação $\mathtt{M}$.

Formalmente:

Uma Rede de Petri pode ser definida como uma 5-tupla $\mathrm{PN} = \{\mathrm{P}, \mathrm{T}, \mathrm{F}, \mathrm{W}, \mathrm{M}_0\}$ onde:

$\mathrm{P} = \{\mathrm{p}_1, \mathrm{p}_2, \ldots, \mathrm{p}_\mathrm{m}\}$é um conjunto finito de $\mathtt{m}$ lugares, com $\mathtt{m} > 0$,

$T = \{t_1, t_2, \ldots, t_n\}$ é um conjunto de $\mathtt{n}$ transições, com $\mathtt{n} > 0$,

$P \cap T = \emptyset$,

$F \subseteq \{P\ X\ T\} \cup \{T\ X\ P\}$ é o conjunto de arcos,

$\mathrm{W}: \mathrm{F} \rightarrow \{1,2,3,..\}$ é a função peso;

$\mathrm{M}_0: \mathrm{P} \rightarrow \{0,1,2,3,..\}$ é a marcação inicial,

$\mathrm{P} \cap \mathrm{T} = \emptyset$ e $\mathrm{P} \cup \mathrm{T} \neq \emptyset$.

Uma rede de Petri sem marcação inicial é denotada por $\mathrm{N} = \{\mathrm{P}, \mathrm{T}, \mathrm{F}, \mathrm{W}\}$.

Regras de ativação:

Uma transição está habilitada se cada arco de entrada possuir um número de marcas maior ou igual a $W(p,t)$;

Uma transição habilitada pode ser ativada ou não, dependendo dos eventos;

Quando uma transição é ativada, ela retira de cada lugar de entrada $W(p,t)$ marcas e adiciona a cada lugar de saída $W(t,p)$ marcas.

Na Figura 2.7 é dado um exemplo de ativação de uma transição, mostrando os estados dos lugares antes e após a transição.

Pode-se ver que os lugares H_2 e O_2 possuem 2 marcas cada, o arco de entrada da transição `t` vindo de H_2 tem peso 2, o arco de entrada da transição `t` vindo de O_2 tem peso 1 (omitido).

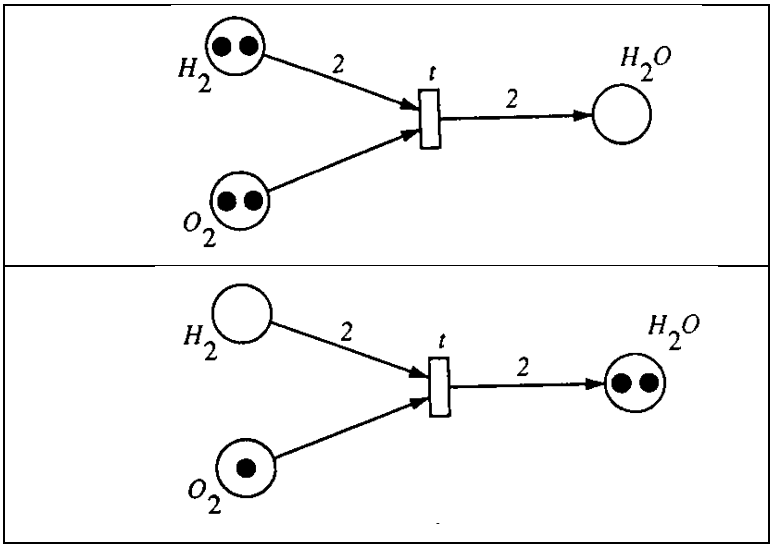

Figura 2.7 Estado da Rede de Petri antes da ativação e após ativação da transição

Vê-se que após a transição ser ativada duas marcas são retiradas do lugar H_2 e uma marca é retirada do lugar O_2, e que o lugar H_2O recebe 2 marcas, que é o peso do arco de saída da transição `t` para o lugar H_2O.

Concorrência

Redes de Petri são apropriadas para modelagem de atividades que exigem decisões ou não determinismo.

Pelo exemplo dado na Figura 2.8 pode-se perceber que existe um conflito, escolha ou não determinismo na ativação da transição t_1 ou t_2 pois depende de algum evento fora da rede que resolva o conflito, tomando a decisão de qual será ativado, podendo inclusive ser uma decisão tomada aleatoriamente. Esta característica da Rede de Petri em muito auxilia a modelagem de problemas em que a concorrência por recursos é uma constante.

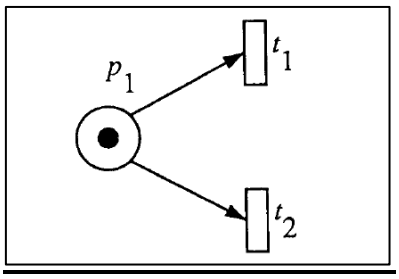

Figura 2.8 Estrutura em Rede de Petri conhecida como conflito, escolha ou decisão

Entre os métodos de análise de uma Rede de Petri, duas são de interesse desta tese: árvore de cobertura e equação de estados.

Árvore de cobertura

Dada uma rede de Petri com marcação inicial, a partir da marcação inicial pode-se obter uma ou mais novas marcações de acordo com as transições habilitadas e ativadas. Para cada nova marcação pode-se obter novas marcações e assim sucessivamente. Este processo gera uma árvore de cobertura cuja raiz é a marcação inicial.

Os nós representam as marcações e os arcos representam as transições ativadas que levam de uma marcação para outra marcação.

Para modelar um problema e traçar estratégias é uma ferramenta poderosa, porém esta árvore tende a crescer na medida em que o número de lugares e transições na rede de Petri aumenta, tornando sua visualização impossível a partir de determinado ponto.

A árvore de cobertura é o espaço de soluções para um problema de PRP modelado com uma Rede de Petri. Pode-se obter uma solução iniciando pelo nó raiz e terminando em algum nó folha.

Um caminho nesta árvore inclui todas as operações que devem ocorrer na programação, na exata ordem em que são ativadas, o que lhe garante alta correlação com *makespan*.

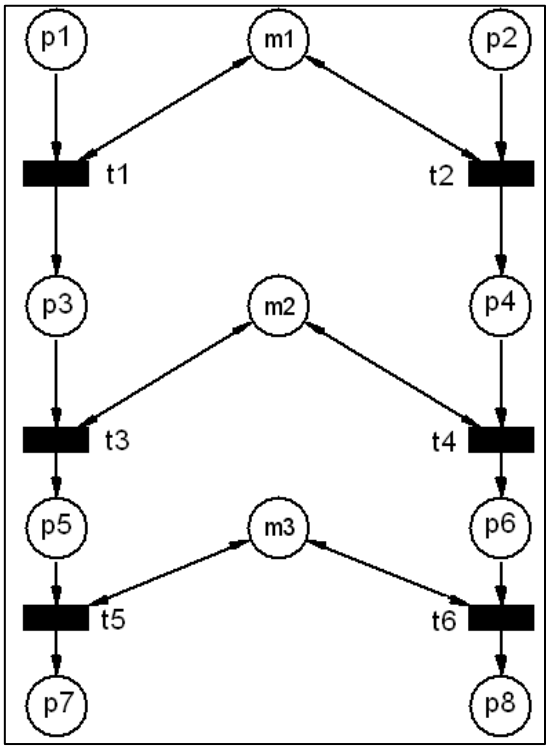

Figura 2.9 Rede de Petri de um job shop.

Exemplo:

Considere a rede de Petri da Figura 2.9. Pode-se perceber que possui 11 lugares (três dos quais compartilhados ou em disputa) e seis transições. Esta rede representa o problema de *job shop* apresentado neste capítulo. Será útil aqui para apresentar a árvore de cobertura.

Nesta rede de Petri pode-se notar que após um número de ativações a rede encerra as ativações. Pode-se imaginar que se esta rede possuísse alguns laços realimentando os lugares p1 e p2, seria uma rede sem fim determinado.

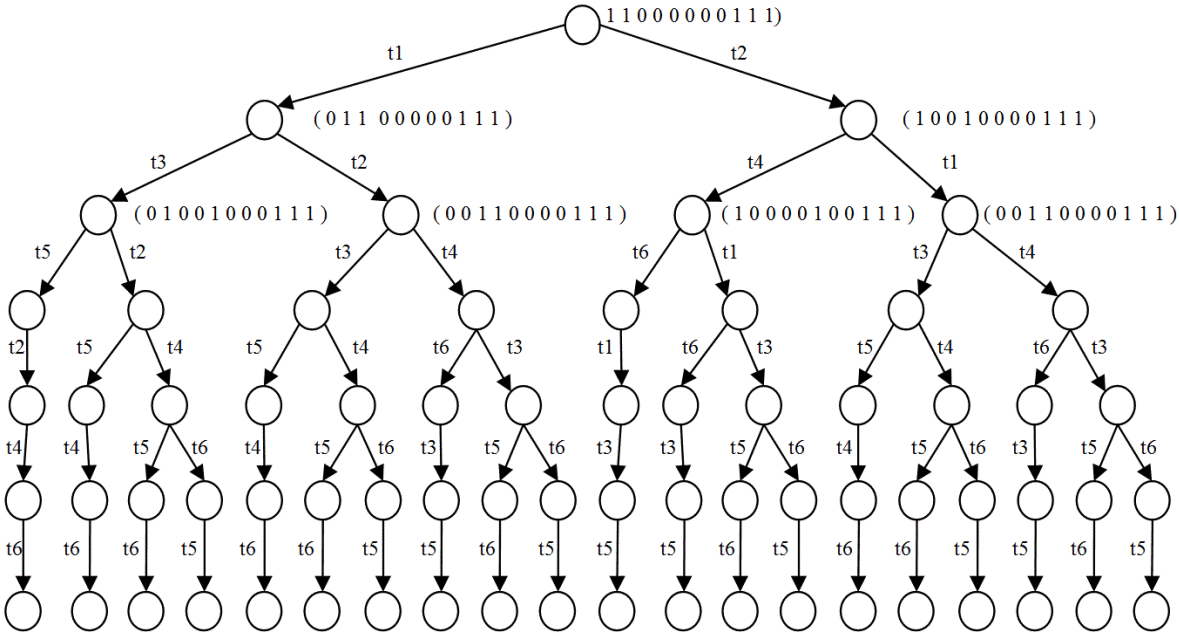

Figura 2.10 Árvore de cobertura total para a Rede de Petri apresentada na Figura 2.9.

Para esta rede, na Figura 2.10 pode-se ver sua árvore de cobertura. As marcações foram mapeadas do seguinte modo (p1 p2 p3 p4 p5 p6 p7 p8 m1 m2 m3).

Percebe-se que embora o número de lugares e transições seja pequeno, a árvore gerada se torna algo complexa, porém ainda útil na visualização de estratégias de trabalho. Pode-se perceber inclusive que as marcações só estão colocadas nos três primeiros níveis de nós da árvore, posto que ficaria ilegível caso se tentasse legendá-la por completo.

Matriz de transição e equação de estados

Considerando a Rede de Petri da Figura 2.9 e sua árvore de cobertura dada na Figura 2.10, percebe-se que a representação em árvore de cobertura começa a ficar difícil.

Nestes casos, o uso da Matriz de transição (ou de incidência) e das equações de estado tornam o trabalho mais produtivo.

A equação de estados é representada abaixo:

$$M_k = M_{k-1} + A^T U_k \, , k = 1,2,3, \ldots .$$

Considere a matriz de transição $A=\{a_{ij}$ é um inteiro, com $i=1,2,..,n$ e $j=1,2,..,m$, sendo n o número de transições e m o número de lugares$\}$. Os elementos a_{ij} da matriz são calculados da seguinte forma: $a_{ij}=w(i,j) - w(j,i)$, em que $w(i,j)$

é o número de arcos que saem da transição `i` e vão para o lugar `j` e `w(j,i)` é o número de arcos que saem do lugar `j` e vão para a transição `i`.

A matriz coluna U_k indica que a `k` transição será ativada. Para os objetivos desta tese, sempre é composta por zeros e apenas um único elemento igual a 1 [MURATA, 1989]. O uso de mais de um elemento diferente de zero implicaria em ativações ocorrendo concorrentemente, o que conflitaria com o objetivo nesta tese de obter uma sequência inequívoca das ativações.

A matriz coluna M_k é a marcação na `k` ativação de uma transição.

A matriz coluna M_{k-1} é a marcação anterior a M_k e que dada a ativação indicada na matriz U_k transforma-se em M_k.

Portanto, a equação de estados permite que se calcule a marcação M_k desde que se conheça sua marcação anterior (M_{k-1}), a matriz de incidência da Rede de Petri (`A`) e a transição a ser ativada (indicada em U_k).

$$M_k \quad M_{k-1} \quad A^t \quad U_k$$

$$\begin{vmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 1 \end{vmatrix} = \begin{vmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 1 \end{vmatrix} + \begin{vmatrix} -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 0 \\ 1 & 0 & -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 & 0 & -1 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{vmatrix} \begin{vmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{vmatrix}$$

Figura 2.11 Equação de estados instanciada para o exemplo dado na Figura 2.8.

Para o exemplo dado na Rede de Petri da Figura 2.9, mapeado na árvore de cobertura da Figura 2.10, segue na Figura 2.11 sua matriz de incidência `A` e a respectiva equação de estados.

Espaço de soluções e Estrutura de Vizinhança na Rede de Petri

O espaço de soluções, quando se usa a Rede de Petri, pode ser visto como sua árvore de cobertura, pois nela pode se encontrar todas as possíveis soluções.

Uma solução é o caminho que leva da raiz até uma das folhas, indicando a sequência de ativações das transições na RP.

A estrutura de vizinhança desta solução é a sequência de tarefas de sua programação, que está na mesma ordem das ativações na árvore de cobertura, sendo portanto altamente correlata ao *makespan*.

2.2.1.2 Os operadores de vizinhança

Dada uma solução, extrai-se a estrutura de vizinhança e a transforma de um vizinho em outro. Os operadores que transformam esta solução em uma solução vizinha são chamados de operadores de vizinhança.

Com o uso da representação da solução baseada em operações (CHENG, 1996) evita-se o problema de a nova solução obtida pela operação na estrutura de vizinhança não ser factível.

Basicamente, há duas formas de se extrair a estrutura de vizinhança:

- A estrutura da vizinhança é a própria solução (RN) ou a sequência de tarefas da programação gerada pela solução (BACO);
- A estrutura da vizinhança é um subconjunto de tarefas (CC) da programação gerada pela solução.

BACO (Baseado em Árvore de Cobertura) é o nome que se deu à estrutura de vizinhança proposta nesta tese e ao trabalho desenvolvido utilizando-a em uma colaboração de AG-VNS.

Após a operação na vizinhança, deve-se retornar a estrutura de vizinhança para a solução e recalcular-se o objetivo. Isto habitualmente ocorre de duas formas:

- A estrutura da vizinhança representa a própria solução (RN e BACO) e basta recalculá-la para obter o novo objetivo;
- A estrutura deve ser reinserida na solução (CC), recalculando-se o objetivo.

Em resumo, recebe-se uma solução. No caso de AG, um cromossomo contendo uma sugestão de sequência de tarefas para a programação. Extrai-se desta solução a estrutura de vizinhança. Opera-se esta estrutura com operadores de vizinhança. Reinsere-se esta estrutura na solução. Recalcula-se a programação, obtendo-se a nova função de avaliação.

Para os exemplos do uso dos operadores, usa-se a solução indicada no exemplo deste Capítulo, que é uma sugestão de sequência de tarefas a ser programada. A solução, a programação obtida e estrutura de vizinhança RN correspondentes são:

- Sugestão de sequenciamento de tarefas (solução):
  (1, 1, 1, 2, 2, 2)
- Programação obtida:
  ((1,1,1,0,30), (1,2,2,30,39), (2,1,1,30,40), (1,3,3,39,44), (2,2,2,40,50), (2,3,3,50,70))
- Estrutura da vizinhança RN desta solução:
  (1,1,1,2,2,2)

A seguir, são apresentados os principais operadores de vizinhança encontrados na revisão.

2.2.1.2.1 O operador *Insert*

O operador *Insert* extrai uma tarefa de uma posição aleatória na estrutura de vizinhança e a reinsere em uma nova posição aleatória (ZOBOLAS; TARANTILIS; IOANNOU, 2009).

Genericamente, considerando a estrutura de vizinhança (o_1, o_2, o_3, o_4, o_5, o_6) e que a posição de extração da tarefa seja a posição 2 e o ponto de reinserção seja a posição 4, obtém-se (o_1, o_3, o_4, o_2, o_5, o_6).

Exemplo:

Usando a estrutura de vizinhança do exemplo dado na Seção 2.2.1.2 e escolhendo aleatoriamente dois pontos para retirada e reinserção da tarefa, neste exemplo posição 2 para retirada e posição 4 para reinserção.

Estrutura da vizinhança RN desta solução com tarefa a sofrer *insert*: (1,1,1,2,2,2)

Retirando-se a tarefa da posição 2 fica assim: (1, 1, 2, 2, 2)

E reinserindo na posição 4 fica assim: (1, 1, 2, 1, 2, 2)

Recalculando-se a programação para esta sequência de sugestões de tarefas:

((1,1,1,0,30), (1,2,2,30,39), (2,1,1,30,40), (1,3,3,39,44), (2,2,2,40,50), (2,3,3,50,70))

Note que, ao se aplicar o operador *Insert*, obtém-se um *makespan* de 70 ut, igual ao *makespan* da programação anterior. Observe que, embora as sugestões de sequência de tarefas sejam diferentes, nesta exemplificação as programações geradas são iguais. Em exemplos de maior dimensão isto é mais difícil de ocorrer, porém, fugiria aos objetivos didáticos deste exemplo.

2.2.1.2.2 O operador *Swap*

O operador *Swap* troca duas tarefas em duas posições aleatórias da estrutura de vizinhança (ZOBOLAS; TARANTILIS; IOANNOU, 2009). Deve-se evitar que a operação de *Swap* aconteça com duas posições que contenham a mesma tarefa, pois então a operação seria nula. Para evitar, basta comparar o conteúdo das posições e, se for necessário, ir selecionando duas novas posições até encontrar um par delas que possua conteúdos diferentes.

Genericamente, considerando a estrutura de vizinhança (o_1, o_2, o_3, o_4, o_5, o_6) e considerando que as posições de troca das tarefas sejam as posições 2 e 4, obtém-se (o_1, o_4, o_3, o_2, o_5, o_6) .

Exemplo:

Usando a estrutura de vizinhança do exemplo dado na Seção 2.2.1.2 e escolhendo aleatoriamente dois pontos para as trocas das tarefas, neste exemplo as posições 2 e 4.

Estrutura da vizinhança RN desta solução com tarefa a sofrer *Swap*: (1,1,1,2,2,2)

Trocando as tarefas das posições fica assim: (1, 2, 1, 1, 2, 2)

Recalculando-se a programação para esta sequência de sugestões de tarefas:

((1,1,1,0,30), (2,1,1,30,40), (1,2,2,30,39), (1,3,3,39,44), (2,2,2,40,50), (2,3,3,50,70))

Note que, ao aplicar-se o operador *Swap* obtém-se um *makespan* de 70 ut, igual ao *makespan* da programação anterior.

2.2.1.2.3 O operador 2-opt

O operador 2-opt inverte uma sequência de tarefas, escolhendo dois pontos aleatórios na estrutura de vizinhança, extraindo este trecho da estrutura, invertendo-o e reinserindo-o na mesma posição em que se encontrava (ZOBOLAS; TARANTILIS; IOANNOU, 2009).

Nota-se que o 2-opt é uma operação que altera a estrutura de vizinhança de forma muito mais radical do que as operações *Insert* e *Swap*, posto que pode até mesmo mudar a sequencialização de todas as tarefas da programação.

Genericamente:

(o_1, o_2, o_3, o_4, o_5, o_6),

Considerando que as posições de início e fim do trecho a ser invertido como as posições 2 e 5:

(o_1, o_2, o_3, o_4, o_5, o_6) invertendo o trecho entre as posições fica (o_1, o_5, o_4, o_3, o_2, o_6).

Exemplo:

Usando a estrutura de vizinhança do exemplo dado na Seção 2.2.1.2 e escolhendo aleatoriamente dois pontos para início e fim do trecho a ser invertido como as posições 1 e 4.

Estrutura da vizinhança RN desta solução com trecho a sofrer 2-opt: (1,1,1,2,2,2)

Invertendo o trecho fica assim: (2, 1, 1, 1, 2, 2)

Recalculando-se a programação para esta sequência de sugestões de tarefas:

((2,1,1,0,10), (1,1,1,10,40), (2,2,2,10,20), (2,3,3,20,40), (1,2,2,40,49), (1,3,3,49,54)).

Note que, ao se aplicar o operador 2-opt, obtém-se um *makespan* de 54 ut, diferente do *makespan* da programação anterior.

2.2.1.3 Apresentação da correlação entre as EVs e o makespan

Nesta seção apresentam-se por meio de exemplos, as representações de cada EV levantada na literatura e sua correlação com o *makespan*.

Seja o exemplo de um cromossomo baseado em operações para o problema apresentado nas tabelas 2.1 e 2.2:

1,1,1,2,2,2

Considere a tupla definida em na seção 2.2.1.1 e a seguinte programação oriunda desta sugestão e:

Programação

{ (1, 1, 1, 0, 30), (1, 2, 2, 30, 39), (2, 1, 1, 30, 40),

(1, 3, 3, 39, 44), (2, 2, 2, 40, 50), (2, 3, 3, 50, 70)}

Que está representada no gráfico de Gantt da Figura 2.12 abaixo.

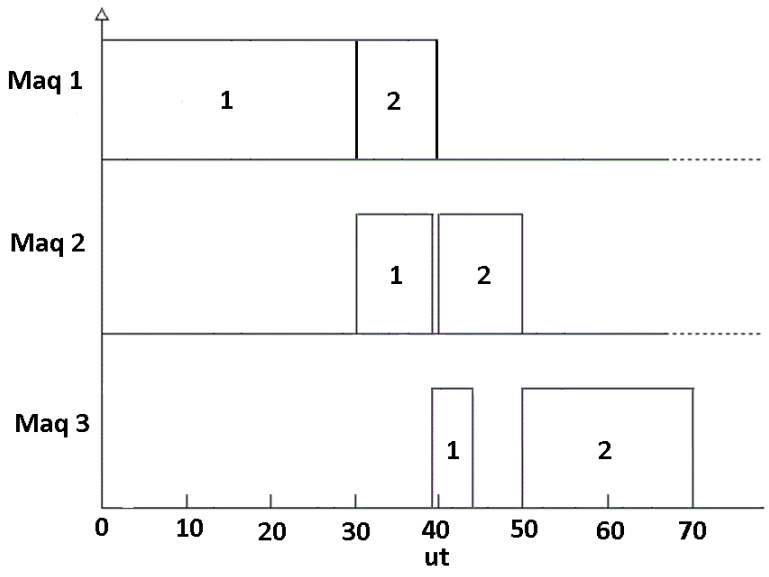

Figura 2.12 Gráfico de Gantt para o exemplo dado.

Como visto na Seção 2.2.1.1.1 a EV RN é : 1,1,1,2,2,2

Como visto na Seção 2.2.1.1.2 a EV CC é: 1,2,2,2

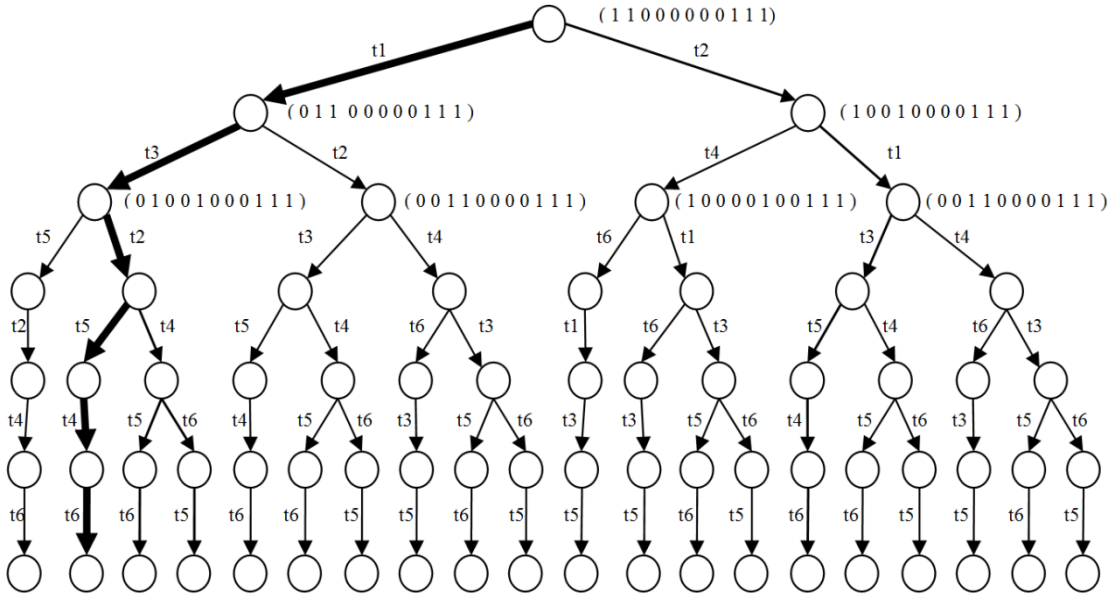

Figura 2.13 Árvore de cobertura para o exemplo dado.

Na Seção 2.2.1.1.3 a EV baseada na AC (Figura 2.13) é: 1,1,2,1,2,2

Analisando as EVs obtidas e a sequência de tarefas, suas fases, a ser produzida pode-se notar que:

1 - Sugestão de solução **1,1,1,2,2,2**

2 - Representação Natural 1,1,1,2,2,2

3 - Caminho Crítico 1,2,2,2

4 - Árvore de cobertura 1,1,2,1,2,2

Programação na ordem dos disparos

{ (1, 1, 1, 0, 30), (1, 2, 2, 30, 39), (2, 1, 1, 30, 40), (1, 3, 3, 39, 44), (2, 2, 2, 40, 50), (2, 3, 3, 50, 70)}

5 - Só as tarefas 1,1,2,1,2,2

Nesse ponto, é interessante notar que a representação típica do sequenciamento em cromossomos (1), que depois transforma-se em programação (5), não possui correlação com o *makespan*. Note-se também que tanto RN (2) como CC (3) não possuem alta correlação com a sequência de tarefas da programação(5).

Por outro lado, a representação da programação (5) usando a sequência de ativações (4) está totalmente correlacionada ao *makespan*.

2.2.1.4 Formas de solucionar o problema de PRP

PRP pode ser solucionado por meio de inúmeras técnicas, indo desde Programação Inteira, Cadeias de Markov, Teoria das Filas, heurísticas e meta-heurísticas (ARENALES et al., 2007). Na próxima seção, duas meta-heurísticas são expostas. No uso de uma nova estrutura de vizinhança aplicada na colaboração entre elas está apoiada esta tese.

2.2.2 **Métodos de Busca**

Existem vários métodos de busca, alguns globais e outros locais. Métodos globais buscam no espaço de busca de forma ampla, tentando achar alguma solução ótima globalmente, e várias soluções boas. Métodos locais buscam achar soluções ótimas locais no entorno das soluções dadas como entrada (AARTS; LENSTRA, 2004).

É possível configurar um método de busca global para atuar como local, agindo em uma região reduzida do espaço de soluções, em torno de alguma boa solução, assim como é possível configurar um método local para agir como um método de busca global, transformando sua vizinhança em algo muito abrangente. Porém, é muito provável que ambas as possibilidades não obtenham bons resultados, pois estariam fazendo uso inadequado das técnicas desenvolvidas para um fim usando-as em outro (AARTS; LENSTRA, 2004).

Muitas vezes, tem sido interessante hibridizar ou colaborar um método global com um local, tirando proveito de ambas as abordagens e obtendo melhores resultados.

Levar em consideração as características do problema, do método a ser usado e configurar adequadamente este conjunto é essencial para que se obtenham bons resultados (AARTS; LENSTRA, 2004).

Especificamente esta tese trabalha com AG e VNS. Estas escolhas e suas configurações foram feitas após o mapeamento indicado mais a frente neste Capítulo.

Nas próximas subseções, são apresentadas as principais características dos métodos de busca global (AG) e busca local (VNS) utilizados nesta tese e constantes em toda revisão da literatura.

2.2.2.1 Algoritmo Genético – AG

Algoritmo Genético é um método de busca global e de otimização com inspiração na evolução dos seres vivos e na genética, apoiando-se na ideia de que, através de cruzamentos e mutações, uma população tende a melhorar, pois os seres mais capazes sobrevivem e os menos capazes não sobrevivem. Os conceitos de Algoritmo Genéticos foram estabelecidos por HOLLAND (1975) e divulgados como um método de uso real por seu aluno GOLDBERG (1989).

Seus principais conceitos, no modo como tem sido usado nos últimos anos na área de PRP em ambientes de *job shop* ou *flow shop*, estão apresentados nesta seção (praticamente todos os artigos constantes na revisão de literatura, seções 2.3.2.1 e 2.3.2.2).

Cromossomo

A estrutura básica do Algoritmo Genético é o cromossomo, que habitualmente carrega uma solução do problema.

As representações comuns em *job shop* se dividem em dois tipos (CHENG, 1996):

- As que fazem uma abordagem direta do problema, codificando alguma forma de sugestão de sequenciamento das operações. O mais comum destas codificações é a representação baseada em operações, que é a usada na maioria das pesquisas (RN) e nesta tese;

- As que fazem uma abordagem indireta, codificando algum conjunto de regras, como as de despacho ou baseadas em máquinas, como orientação para a programação. Esta forma também tem muitos seguidores (CC).

Nesta tese e na maioria das pesquisas encontradas, a representação baseada em operações é usada. Genericamente, considere o cromossomo $(o_1, o_2, o_3 \ldots, o_n)$ onde o_i indica uma sugestão de operação de alguma fase de uma tarefa e n é o número total de operações do PRP.

Exemplo de um cromossomo instanciado seria (2, 1, 3, 1, 2, 2, 3, 1, 3) representando uma sugestão de solução para um problema de três tarefas e três máquinas.

Outro exemplo seria a própria solução usada na seção anterior: (1, 1, 1, 2, 2, 2).

População

Algoritmo Genético trabalha com um conjunto de cromossomos, ao qual se dá o nome de população. A cada iteração do algoritmo, chamada de geração, a população é renovada, total ou parcialmente, dependendo das configurações assumidas.

Função de avaliação – *fitness*

Cada cromossomo representa uma possível solução ao problema dado e possui associado a si um valor que indica sua qualidade. Nesta tese, a métrica é o *makespan*, então, a função de avaliação é dada pela inversa do *makespan*, indicando que quanto menor o *makespan*, maior o *fitness* da solução. A equação para a função de avaliação é:

$$F(x) = \frac{1}{mkp},$$

Onde x é a sugestão de solução constante no cromossomo e mkp é o valor de *makespan* obtido para esta solução.

Elitismo

Elitismo é uma operação que permite manter certo número de cromossomos da população atual na seguinte, como forma de garantir que não se perca todas as boas soluções. Escolhe-se certo número de melhores cromossomos e transfere-se para uma população nova. Pode ser de um cromossomo a alguns.

Seleção

Seleção é uma operação que permite escolher alguns cromossomos da população atual, de forma a poder salvá-los na próxima população ou então operá-los com os operadores de cruzamento e mutação. A quantidade a ser selecionada depende dos parâmetros dados.

A seleção pode se dar por Roleta, Torneio ou outro método conhecido ou descrito pelo pesquisador. Comumente usa-se Roleta, Torneio e Randômica.

Roleta: cada cromossomo assume uma posição na roleta com área igual a uma porcentagem relativa à sua aptidão em relação à soma de todas aptidões da população atual. Roda-se a roleta e seleciona-se o cromossomo sorteado. Repete-se o processo o número de vezes especificado no parâmetro.

Torneio: sorteiam-se alguns cromossomos aleatoriamente da população atual e escolhe-se o melhor entre eles. Usualmente, varia de 3 a 5 candidatos por torneio. Repete-se o processo até se obter todos os cromossomos indicados no parâmetro.

Randômica: os cromossomos são escolhidos por sorteio até se completar a população requerida.

Nesta tese, usa-se uma forma alternativa de seleção buscando tornar a operação de cruzamento mais intensa como busca local, separando-se a população atual em duas populações: `Melhores` e `Piores`. Em `Melhores`, ficam os melhores cromossomos e, em `Piores`, os piores cromossomos. De forma genérica, considerando uma população de `n` cromossomos, e dividindo-a em duas partes `n1` e `n2` talque `n1 + n2` somam `n` e nem `n1` nem `n2` são iguais a zero, e considerando que a população esta ordenada pela ordem crescente de seus makespans, a população chamada de `Melhores` conteria os cromossomos indo do cromossomo 1 ao cromossomo `n1` e a população chamada `Piores` conteria os cromossomos indo do cromossomo `(n1 + 1)` ao cromossomo `(n1 + n2)`.

Operação de cruzamento

A operação de cruzamento busca por meio da mistura de dois cromossomos gerar de um a dois novos cromossomos com qualidade melhorada. Nem sempre se consegue a melhoria, mas comumente mantém-se o novo cromossomo como forma de diversificação.

Seleciona dois cromossomos (pais) da população (ou das populações como nesta teste) e cria-se de um a dois novos cromossomos (filhos) através de alguma técnica de cruzamento.

Nesta operação, deve-se tomar cuidado para que os novos cromossomos sejam factíveis.

Para o problema de PRP nos ambientes de *job shop* e *flow shop*, as operações de cruzamento mais comuns são:

- **PMX – *Partially Mapped Crossover***

  Segundo proposto por Goldberg e Lingle (1985), dados dois cromossomos, faz-se dois cortes aleatórios em ambos e transmite as seções entre eles integralmente para os filhos. Os genes das duas seções copiadas são, então, relacionados um a um e copia-se os genes que sobraram em cada cromossomo pai para o outro cromossomo filho, substituindo os genes pelos relacionados.

  Exemplo:

  P1 2 4 1 3 5 6 -- F1 x x 1 3 5 x

  P2 5 3 1 4 6 2 -- F2 x x 1 4 6 x

  Criam-se as relações 1 e 1, 3 e 4, 5 e 6

  Copia-se os genes de fora da seção do P1 para F2, substituindo os genes relacionados. Copia-se os genes de fora da seção do P2 para o F1, substituindo os genes relacionados.

  Os genes 2, 4 e 6 do P1 são copiados para F2 como 2, 3 e 5; e os genes 5, 3 e 2 do P2 são copiados para F1 como 6, 4 e 2.

  Ficando o cruzamento assim:

  P1 2 4 1 3 5 6 -- F1 6 4 1 3 5 2

  P1 5 3 1 4 6 2 -- F2 2 3 1 4 6 5

- **OX – *Order Crossover***

Segundo proposto por Davis (1985), dois cortes e as subcadeias são copiados integralmente de cada pai para cada filho (pai 1 para filho 1 e pai 2 para filho 2).

Preenchem-se os genes vagos, do final para o começo, com os genes do pai que não doou a subcadeia, excluindo-se os que estão na subcadeia.

P1 2 4 1 3 5 6 -- F1 x x 1 3 5 x

P2 5 3 6 4 1 2 -- F2 x x 6 4 1 x

Sequências a serem usadas em cada filho, vindas de P2 6, 4 e 2 e de P1 2, 3 e 5,

Ficando

P1 2 4 1 3 5 6 -- F1 2 4 1 3 5 6

P2 5 3 6 4 1 2 -- F2 5 3 6 4 1 2

- **CX – *Cycle Crossover***

Segundo proposto por Oliver (1987), cria sequências de genes com mesmos valores em ambos os pais e as copiam para os filhos, e completam os filhos com os genes que sobram dos pais opostos.

P1 2 4 1 3 5 6 -- F1 x x 1 3 5 x

P2 5 3 6 4 1 2 -- F2 5 3 x x 1 x

Copiam-se os genes 1, 3 e 5 nas mesmas posições em que se encontram nos pais para os filhos. Completa-se cada filho com os genes do pai oposto não pertencentes à sequência.

P1 2 4 1 3 5 6 -- F1 6 4 1 3 5 2

P2 5 3 6 4 1 2 -- F2 5 3 2 4 1 6

- **LOX - *Linear Order Crossover***

Segundo proposto por Falkenauer e Bouffouix (1991), faz-se dois cortes, exclui os genes com os valores iguais aos cortes de cada pai no outro pai. Arrumam-se os genes restantes em cada pai de modo a deixar a região entre os cortes vazia. Copia os genes que eram de cada região do corte para os filhos.

P1 2 4 1 3 5 6 → F1 2 x 1 3 5 x – 2 1 x x 3 5 -- 2 1 6 4 3 5

P2 5 3 6 4 1 2 → F2 5 x 6 4 x 2 – 5 6 x x 4 2 -- 5 6 1 3 4 2

- **PPX – *Precedence Preservative Crossover***

Segundo proposto por Mattfeld e Bierwirth (2004), cria-se uma máscara binária, copiando os genes de cada pai para um filho, nas respectivas posições, quando a máscara indicar 0 para o pai 1 e 1 para o pai 2. Elimina-se de cada pai os genes que foram copiados no pai oposto, na mesma ordem em que aparecem na máscara. Completam-se os filhos com os valores faltantes na mesma ordem em que os genes aparecem no pai oposto.

Máscara 0 0 1 0 1 1

P1 2 4 1 3 5 6 -- F1 2 4 x 3 x x sequência do P2 5, 6 e 1 – 2 4 5 3 6 1

P2 5 3 6 4 1 2 -- F2 x x 6 x 1 2 sequência do P1 4, 3 e 5 -- 4 3 6 5 1 2

Esta é a forma de fazer o cruzamento usada nesta tese, usando uma máscara com a primeira metade com o número zero e a segunda metade com o número um.

Operação de Mutação

Esta operação transforma um cromossomo pela mudança de posição de alguns genes ou alguma outra técnica. A mutação busca evitar a estagnação do AG, inserindo cromossomos que fogem da convergência prematura. Comumente, usa-se selecionar aleatoriamente um ou mais cromossomos da população intermediária. As mutações, normalmente, ocorrem na população intermediária mas em algumas pesquisas ocorrem na nova população gerada. O número de cromossomos a sofrer mutação é indicado nas configurações do AG.

Entre as operações comuns de mutação estão: *swap*, *insert*, 2-*opt*. Estes operadores foram explicados na seção de operadores de vizinhança e, diferentemente de quando são aplicados em uma estrutura de vizinhança, sua aplicação na representação do cromossomo costuma levar a solução para outro ponto de espaço de soluções, muito provavelmente, distante do ponto de origem.

Exemplo de aplicação do operador *swap* (posições 3 e 6):

2 4 1 3 5 6 → 2 4 6 3 5 1

Uma mutação comum é a geração de um cromossomo com novos genes, escolhidos aleatoriamente.

Exemplo de mutação randômica:

2 4 1 3 5 6 → 3 1 5 2 6 4

Complemento de população

Quando se gera uma nova população, pode acontecer de esta possuir um número de cromossomo menor que o número total da população desejada, resultado de se juntar os cromossomos vindos da elite e da seleção na qual ocorreram cruzamento e mutação. Neste caso, faz-se uma nova seleção com um número de cromossomos igual ao necessário para complementar a nova população.

Por exemplo: para uma população de trinta elementos, elite de um elemento, taxa de cruzamento de 66% (20 cromossomos), a população complementar seria de nove cromossomos.

Condição de parada

Um Algoritmo Genético costuma ficar produzindo gerações até que se satisfaça alguma condição de parada. Habitualmente usa-se o número de gerações, o número de gerações sem melhora na solução, o tempo de processamento, ou outra forma definida pelo pesquisador.

Nesta tese, o foco é PRP, então, o limite de tempo de processamento é sempre a forma escolhida.

Parâmetros do AG genérico

Os principais parâmetros e configurações de um Algoritmo Genético são:

Tamanho da população: número de cromossomos que compõem a população;

Forma de geração da população inicial: aleatória ou outra forma;

Número de cromossomo da Elite: número de cromossomos copiados para a nova população;

Tipo de seleção: Roleta, Torneio ou outro tipo;

Taxa de cruzamento: quantos cromossomos serão atualizados por cruzamento;

Tipo do cruzamento: qual o método usado no cruzamento;

Taxa de mutação: quantos cromossomos serão atualizados por mutação;

Operador de mutação: que operador será utilizado na mutação;

Condição de parada: qual condição será usada.

Passo a passo de um Algoritmo Genético

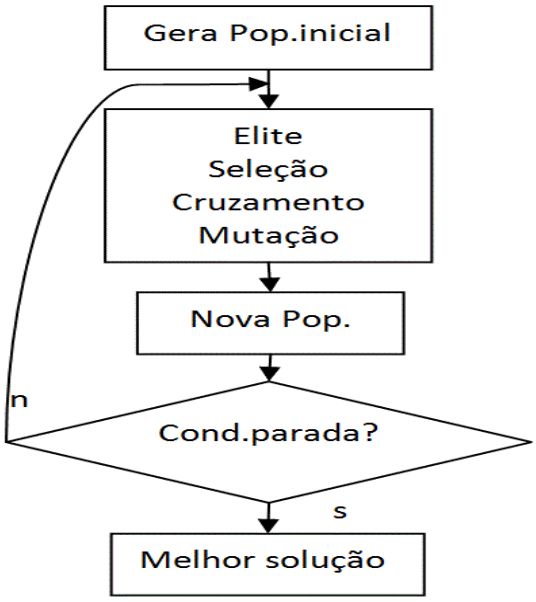

Figura 2.14 Diagrama genérico de AG (Fonte:autor)

Basicamente, um Algoritmo Genético segue os seguintes passos:

- Passo 1 – gerar uma população atual (Inicial), normalmente de forma aleatória;
- Passo 2 – Avaliar toda população atual;
- Passo 3 – Gerar uma nova população pelos seguintes subpassos:
- Passo 3.1 – Copiar, da população atual, para a população nova o número de cromossomos indicado pelo parâmetro elite;
- Passo 3.2 – Selecionar da população atual o número de cromossomos indicado pela taxa de cruzamento para a população intermediária;
- Passo 3.3 - Aplicar cruzamento nos cromossomos desta população intermediária, substituindo os cromossomos anteriores pelos novos;
- Passo 3.4 - Aplicar mutação nos cromossomos desta população intermediária;
- Passo 3.5 - Copiar a população intermediária para a população nova;
- Passo 3.6 – Selecionar, da população atual, o número de cromossomos necessários para completar o tamanho da população nova e copiá-los para esta;
- Passo 4 - Transformar a população nova na população atual;
- Passo 5 - Avaliar toda a população atual;
- Passo 6 - Se a condição de parada for satisfeita seguir para o Passo 7, se não, ir para o Passo 3;
- Passo 7 - Retornar a melhor solução encontrada.

2.2.2.2 Variable Neighborhood Search – VNS

Variable Neighborhood Search (VNS – Busca em Vizinhança Variável) é uma meta-heurística de busca local proposta por (MLADENOVIC e HANSEN, 1997; MLADENOVIC, HANSEN e PEREZ, 2010) e baseia-se na ideia de fazer a busca local trocando as vizinhanças. Usualmente, estas vizinhanças são interpretadas como operadores de vizinhança.

Normalmente, VNS é utilizado em uma colaboração ou hibridização com um método de busca global.

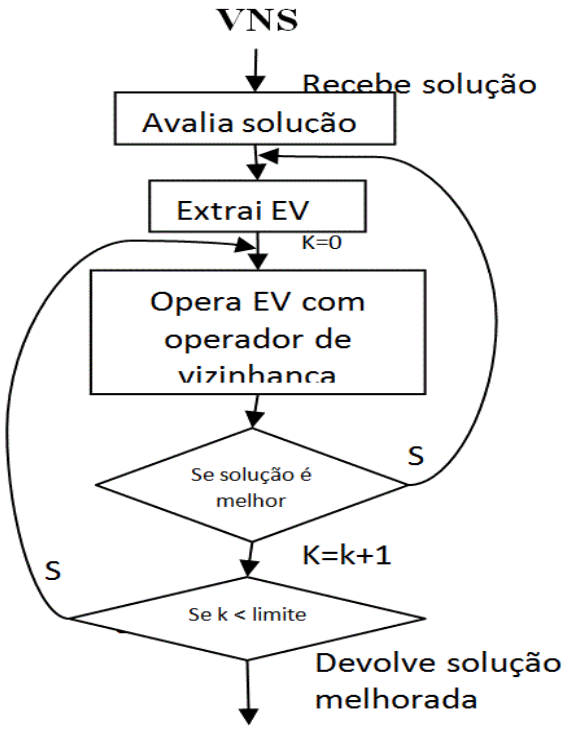

Figura 2.15 Diagrama do VNS genérico (Fonte: autor).

Genericamente, os passos de um VNS são:

- Passo 1 – Definir quais operadores de vizinhança serão usados, fazendo limite = número de operadores;
- Passo 2 – Receber como entrada uma solução;
- Passo 3 – Calcular a qualidade desta solução. Guardar a solução e sua qualidade como a melhor encontrada;
- Passo 4 - Extrair a estrutura de vizinhança desta solução;
- Passo 5 – Fazer k = 1;
- Passo 6 - Aplicar o k-operador de vizinhança na estrutura de vizinhança;
- Passo 7 – Reinserir a estrutura de vizinhança na solução;
- Passo 8 – Calcular a qualidade da nova solução;
- Passo 9 – Comparar a qualidade da nova solução com a melhor encontrada.
  Se a qualidade for melhor do que a melhor encontrada, substituir a melhor encontrada pela nova e voltar ao Passo 6;

Se a nova qualidade for igual ou inferior do que a melhor encontrada, faz k = k + 1. Se k <= limite, então, voltar ao Passo 6, se não encerrar a busca local e retornar a melhor encontrada.

Solução inicial

A entrada é uma solução passada por um método global ou então gerada por algum processo quando o VNS atua sozinho como método de otimização. Quando o método trabalhando junto com o VNS é um AG, a solução costuma ser passada na forma de cromossomo.

Extração de estrutura de vizinhança

Como busca local, o VNS aplica os operadores em uma vizinhança da solução dada, sendo necessário que ocorra uma extração de tal vizinhança anteriormente.

Esta extração está explicada na seção anterior, quando se expõe as estruturas de vizinhança e seus algoritmos de extração. Relembrando: a estrutura de vizinhança RN é idêntica ao cromossomo passado; a estrutura de vizinhança CC precisa ter a programação obtida gerada para o cromossomo de entrada, então sendo extraída desta programação; e a estrutura de vizinhança desta tese é extraída da programação obtida do cromossomo de entrada (a ser explicado sucintamente no Capítulo 3).

Reinserção da estrutura de vizinhança

Uma vez que a estrutura de vizinhança foi operada, deve-se retorná-la para a solução, de forma a se poder recalcular a qualidade desta solução alterada.

Os algoritmos de reinserção estão explicados na seção anterior quando se explica as estruturas de vizinhança. O algoritmo de reinserção da estrutura de vizinhança desta tese é explicado no Capítulo 3.

Condição de parada

Na revisão de literatura a condição de parada mais aplicada é a de alcançar um número fixo de aplicações de operadores na solução. Outras condições são: tempo de processamento, primeira melhoria encontrada, alcançar um número de aplicações sem melhorias, entre outras.

Muitas variações são propostas nas pesquisas atuais, sendo que a variação mais utilizada é a em que se executa cada operador um número fixo de vezes e só, então, troca-se de operador.

Operadores de Vizinhança

Na revisão de literatura, percebe-se que os operadores mais usados são *Insert*, *Swap* e *2-opt*.

2.3 Artigos correlatos

Nesta seção está o mapeamento das pesquisas feitas na área de PP e PRP e a revisão da literatura atual, assim como a análise dos dados relevantes encontrados.

2.3.1 Mapeamento das pesquisas em PP e PRP buscando minimizar *makespan*

Esta revisão dá-se através do portal CAPES e é feita uma pesquisa usando a cadeia de busca "*makespan AND (reactive OR rescheduling OR reprogramming OR neighborhood)*" em todas as bases da aba Ciências da Terra - Ciência da Computação, sendo que destas, nove foram selecionadas por retornarem resultados. A cadeia de busca permite encontrar os artigos que tratam de *makespan* e/ou de busca local ou de PRP. A seleção dos artigos interessantes fica a cargo do pesquisador.

Como se pode ver na Tabela 2.3, o número de artigos retornados pela consulta é relevante, e percebe-se que, nos últimos anos, a pesquisa mostra-se ainda pujante, considerando que as primeiras pesquisas datam de antes de1990.

Tabela 2.3 Busca Portal CAPES na área de Computação, string de busca "*makespan AND (reactive OR rescheduling OR reprogramming OR neighborhood)*", todas as bases que possuem textos completos (última atualização em 21/01/2015).

Base	Total	<2010	2010
Academic Search Premier - ASP (EBSCO)	73	23	50
ACM Digital Library m&n	138	59	79
Cambridge Journals Online	66	32	34
IEEE Xplore	5	2	3
Oxford	5	3	2
ScienceDirect (Elsevier)	1964	962	1002
SpringerLink (MetaPress)	1073	482	591
Wiley Online Library	278	171	107
World Scientific	86	50	36
TOTAIS	3688	1784	1904

A revisão se dá em artigos que buscam resolver o problema de Programação da Produção ou Programação Reativa da Produção, o qual é referenciado ao longo desta revisão como simplesmente programação, a menos que seja indicado o contrário.

Os artigos tratando especificamente de programação da produção são utilizados para se obter informações sobre métodos e parâmetros, pois o interesse desta tese está em PRP e não em PP.

Na Tabela 2.4 tabula-se o número de artigos por publicação.

Tabela 2.4 Artigos por publicação que entraram nessa revisão.

Publicação	Número de artigos
Computers & Operations Research	1
Int J Adv Manuf Technol	2
Future Generation Computer Systems	1
European Journal of Operational Research	1
Expert Systems with Applications	1
Information Sciences	1
Computers & Industrial Engineering	1
J heuristics	1
Procedia Engineering	1
ICNC	1
Outros	2
Total	13

O problema de programação é muito pesquisado nos dias atuais e o objetivo de minimizar o *makespan* é um dos mais almejados, considerando-se o número de publicações a respeito nos últimos anos (vide Tabela 2.3).

Pode-se exemplificar seu uso tanto em manufaturas quanto em atribuição de *jobs* aos processadores, onde pode-se ver claramente a importância do fator tempo de resposta na criação da programação da produção, assim como o *makespan*.

Os trabalhos pesquisados variam nos seus problemas, mas todos possuem em comum a tentativa de programar recursos compartilhados utilizando o conceito de busca na vizinhança visando minimizar o *makespan*.

A revisão propriamente dita, recuperando as informações dos métodos globais e locais utilizados, estrutura da vizinhança da solução, operadores de busca na vizinhança e as bases dos cenários utilizados na validação, acontece somente nos artigos publicados desde 2010 até a data do fechamento desta tese.

Convém ressaltar que a grande maioria dos artigos fazem suas validações em cenários de vários portes. Estes cenários ou são gerados a partir de especificações contidas em referências indicadas ou extraídas diretamente de sítios na *Internet*. Nos resumos dos artigos a referência é apontada, sem porém mencionar como os cenários são obtidos.

2.3.2 **Revisão das pesquisas usando AG e VNS**

Nesta seção são apresentados os artigos que norteiam algumas escolhas desta tese e reforçam outras. São divididos em dois grupos: um com os artigos que usam a estrutura de vizinhança Representação Natural; outro com os artigos que usam a estrutura de vizinhança Caminho Crítico. Os artigos usados na revisão vem do mapeamento (seção anterior).

Sempre que é possível e relevante, para cada artigo é levantado: problema, motivação, justificativa, configurações e uma crítica ao trabalho mostrando onde está o *gap* ou a inspiração das escolhas, dizendo quais resultados obtém-se. Complementando-se a crítica é citado o que se aproveita do artigo na tese.

Entre as características levantadas estão: problema abordado, estrutura do cromossomo, taxa e tipo de cruzamento, taxa e tipo de mutação, tipo de vizinhança. operadores de vizinhança, bases de cenários e método de validação.

Três observações sobre esta revisão:

- A maioria dos artigos usa CC ou RN como estrutura de vizinhança, ficando claro que a falta de correlação destas estruturas de vizinhança com o *makespan* possivelmente deve ter influenciado seus resultados, embora todos os artigos afirmem que tiveram bons resultados;
- A grande maioria dos artigos usa a comparação dos Erros Relativos (ER) como forma de validação, o que não garante grande qualidade, pois sem ferramentas estatísticas adequadas e com ER próximos, qualquer afirmação de superioridade de um método fica comprometida;
- Em vários artigos existe a combinação de dois métodos, sendo que um não é o AG ou o VNS. Nesses casos, uma rápida explicação do método é feita, mas sem intuito de se aprofundar, visto que estes métodos não fazem parte da proposta, bastando ter-se uma vaga ideia de como trabalham.

As próximas seções relatam os artigos em duas categorias e os analisam como uma visão geral.

2.3.2.1 Artigos usando a Estrutura de Vizinhança Representação Natural

Roshanaei et al. (2009) buscam resolver o problema de *job shop* (com dependência da sequência de tempos de setup) usando VNS e tempo limitado para o cálculo da programação. Trabalham com 3 variações do operador de vizinhança *insert*, sendo uma a normal com 1 operação, e as outras duas com 2 e 3 operações respectivamente. Validam o trabalho com cenários vindos de (TAILLARD, 2015), comparando os erros relativos com de outros 4 trabalhos vindos da literatura, testando sua validade usando ANOVA.

Embora seja um algoritmo de busca local, a estrutura de vizinhança RN usada não possui correlação com a programação recebida como entrada. Este trabalho ajuda a reforçar na proposta da presente tese a ideia do uso do método de busca local VNS, da representação do cromossomo como baseado em operações, do operador de vizinhança *insert*, da base de cenários (TAILLARD, 2015), do uso do problema de *job shop* para validação.

Em Jarboni, Eddaly e Siarry (2011) hibridiza-se o método de busca global Algoritmo Genético com o método de busca local VNS para tratar do problema de *No-Wait Flow Shop*, problema comum em indústrias como as químicas e farmacêuticas. A população é constituída de 10 cromossomos, do tipo baseada em operações e usam o mecanismo de roleta como forma de seleção no AG. A mutação ocorre pelo uso do operador *insert* e são usadas taxas de 100% no cruzamento e 80% na mutação. Na busca local aplicou-se *insert* e *swap*. Criam um operador de cruzamento para o AG chamado "*Block Order Crossover*", especificamente pensado para o problema. Validam em cenários de diversos tamanhos vindos de (REEVES, 1995), (HELLER, 1960) e (TAILLARD, 2015). Seus resultados apontam que a hibridização obtém melhor performance que os concorrentes, porém se baseia apenas nos erros relativos das médias por cenário. A vizinhança é a RN que se sabe não ser correlata a programação obtida. Reforça a ideia de trabalhar AG e VNS juntos.

Zobolas, Tarantilis e Ioannou (2009) hibridizam Algoritmo Genético (AG) com Busca em Vizinhança Variável (VNS) para tentar solucionar o problema de programação de Flow Shop Permutacional, criando a população inicial por várias heurísticas, otimizando com os operadores do AG e aprimorando com o uso de 4

operadores (*Swap, Insert, 2-opt, Or-opt*) no VNS. Usam representação do cromossomo baseado em tarefas, A estrutura da vizinhança é a RN e a proposta foi validada com cenários de (TAILLARD, 2015). Interessante pela fartura de ilustrações e resultados, servindo como inspiração para a colaboração AG e VNS.

Song e Xu (2010) propõem hibridizar Algoritmo Genético com *Chaos Local Search* para solucionar a programação de *Job Shop* Flexível, gerando a população inicial em parte aleatoriamente, fazendo a busca na estrutura da vizinhança RN com operador *Insert* , taxa de cruzamento de 70% e taxa de mutação de 2%. Validam em cenários vindos de um sítio da *internet*.

Chaotic Local Search (CLS) (CHOI; LEE, 1998) é um método de busca local baseado em múltiplas mudanças na solução, escolhidas aleatoriamente, definindo-se como CLS de x variáveis. Não se aprofunda nesta descrição por não pertencer ao escopo da tese.

Sun et al. (2010) hibridizam Otimização por Enxame de Partículas (PSO) com Busca em Vizinhança Variável (VNS) para tratar do problema de programação de *Flow Shop* Permutacional. Nesta abordagem usam população inicial gerada em parte pela heurística Nawaz-Enscore-Ham - NEH (NAWAZ; ENSCORE; HAM, 1983) e em parte aleatoriamente; usam a estrutura de vizinhança RN, aplicando o operador *Insert* de duas formas distintas na busca. Validam em cenários vindos de Taillard baseando-se em erro relativo nas comparações. Artigo interessante por hibridizar PSO com VNS, usar Taillard (TAILLARD, 2015) e codificação da solução na forma de chaves, além de iniciar a população com outra forma diferente da aleatória, embora para fins de prova de método, a forma de iniciar a população não influencia os resultados.

PSO (KENNEDY; EBERHART, 1995) é um método de busca global com inspiração no vôo das aves marítimas, sobrevoando o oceano e indicando onde se encontram bons cardumes. Começa com uma população inicial e a cada iteração atualiza os indivíduos aproximando-os do melhor indivíduo encontrado na iteração anterior.

Em Pan et al. (2011b) é proposta a hibridização do método de busca global *Discrete Differential Evolution Algorithm* (DDEA) com o método de busca local

Busca em Vizinhança Variável (VNS) para encontrar solução de problema de programação de *Flow Shop* com buffers intermediários. A estrutura da vizinhança é a RN e a população inicial obtida pelo uso da heurística Nawaz-Enscore-Ham (NAWAZ; ENSCORE; HAM, 1983), sendo que a busca na vizinhança ocorre pelo uso dos operadores *Insert* e *Swap*. A validação é feita utilizando cenários gerados a partir das indicações em (TAILLARD, 2015) usando como comparação os erros relativos.

DDEA (STORN; PRICE, 1997) é um método de busca global de inspiração evolutiva, possuindo população inicial, operação de cruzamento e mutação, com uma variante de excluir os novos cromossomos caso não melhorem, entre outras características.

Yazdani, Amiri e Zandieh (2010) utilizam processamento em paralelo para tratar do problema de programação para *Job Shop* Flexível, usando o *Parallel Variable Neighborhood Search* (PVNS), gerando população inicial pela heurística Pezzella indicada no texto, usando estrutura de vizinhança RN e operando na base de trocas de posições no vetor. A validação se dá em cenários vindos dos sítios na *internet:* Kacem (KACEM; HAMMADI; BORNE, 2002a; KACEM; HAMMADI; BORNE, 2002b), BRdata (BRANDIMARTE, 1993), DPdata (DAUZÉRE-PÉRÉS; PAULLI,1997) e HUdata (HURINK; JURISCH;THOLE, 1994), comparando os erros relativos.

Zheng e Yamashiro (2010) propõem o uso de *Quantum Differential Evolutionary Algorithm* (QDEA) para resolver programação de *Flow Shop*, com população inicial aleatoriamente gerada e usando como busca local o Busca em Vizinhança Variável, operando na estrutura de vizinhança RN com os operadores *Insert, Swap* e *2-opt*. Validam em cenários vindos de várias bases, incluindo Taillard. Validação comparando os erros relativos.

QDEA (K-H, 2000) é um método de inspiração biológica, muito assemelhado ao AG, usando variáveis quânticas para ponderar as alterações.

Em Tang et al. (2011) propõe o uso de hibridização de Algoritmo Genético com Chaos PSO para resolver o problema de *Job Shop* Flexível. Usa o cruzamento POX com taxa de 85% e mutação por *insert* com taxa de 15%. O trabalho se valida

com as bases vindas de Kacem indicadas no texto, comparando os desvios relativos, que são uma variação do erro relativo.

2.3.2.2 Artigos usando a Estrutura de Vizinhança Caminho Crítico

Em Sels, Craeymeersch e Vanhoucke (2011) são apresentados duas configurações de hibridização: Algoritmo Genético com VNS e *Scatter Search* com VNS com duas populações. O trabalho busca comparar os resultados obtidos para o problema de j*ob shop*, entre si e com outros trabalhos. Os autores acreditam que o uso de duas populações no Scatter Search (GLOVER, 1994) garante maior balanceamento entre a diversificação e a intensificação da busca, porém pelos números apresentados fica claro que as configurações dadas só são superiores quando comparadas com as médias entre os trabalhos publicados, obtendo os piores resultados quando executados nos cenários vindos de (TAILLARD, 2015). Como método de validação se baseia nas médias dos erros relativos em relação aos melhores resultados conhecidos por base. Embora use uma busca local, percebe-se que a estrutura de vizinhança usada (CC) não é totalmente correlata à programação obtida. Este trabalho reforça as seguintes escolhas nesta tese: da base Taillard, da representação de cromossomo baseada em operações, do VNS como busca local, do AG como busca global, hibridização de AG com VNS, dos operadores de vizinhança *insert* e *swap*, uso do mecanismo de roleta como forma de seleção no AG, alta taxa de cruzamento (90%) com uso de PPX e taxa de mutação em 10% com uso operação de inversão.

Wang e Zhang (2011) hibridizam Algoritmo Memético com a busca local Busca em Vizinhança Variável para tratar do problema de programação para *job shop*. Usam cromossomo do tipo baseado em operações e a estrutura de vizinhança Caminho Crítico com os operadores *insert* e *swap* na busca local. Geram uma população inicial aleatória de tamanho dependente do cenário, indo de 100 a 500 cromossomos, taxa de cruzamento de 80% com OX e taxa de mutação de 1% com *swap*. Com critério de parada em 150 gerações, validam em cenários vindos de OR-library e obtêm bons resultados comparando as suas médias das execuções em vários cenários com a de outros trabalhos vindo da literatura. Percebe-se que a estrutura de vizinhança CC embora possuindo certa correlação com a programação obtida, não a possui em caráter

total e que o método da validação se baseia apenas nas médias das execuções, não tendo sido feito cálculos estatísticos para garantir a representatividade destas amostras.

Vela, Varela e González (2010) hibridizam Algoritmo Genético com a busca local Busca em Vizinhança Variável para tratar do problema de programação para J*ob Shop* com sequência dependente de tempos de configuração. Usam cromossomo do tipo baseado em operações e a estrutura de vizinhança Caminho Crítico com os operadores *insert, swap e 2-opt* na busca local. Usa o operador CX no cruzamento, embora o tenha nomeado de outra forma, e variam tanto as taxas de cruzamento e mutação como a forma de operar a mutação. Validam em cenários vindos de várias bases comparando os erros relativos.

Gao et al. (2011) hibridizam Algoritmo Memético com a busca local Busca em Vizinhança Variável para tratar do problema de programação para *job shop*. Usam cromossomo do tipo baseado em operações e a estrutura de vizinhança Caminho Crítico com os operadores *insert* e *swap* na busca local. Geram uma população inicial aleatória de tamanho dependente do cenário, indo de 100 a 500 cromossomos, taxa de cruzamento de 80% com CX e taxa de mutação de 1% com *swap*. Validam em cenários vindos de OR-library (BEASLEY, 1990).

2.3.2.3 Análise da Revisão de Literatura

Nesta seção apresenta-se uma análise dos artigos pesquisados, na forma de uma panorâmica da área.

Na Tabela 2.5 estão listadas as principais características das abordagens levantadas na revisão de literatura, e na última linha, as características da abordagem desta tese, nomeada BACO (Busca em Árvore de CObertura).

Nesta revisão fica claro que a PP/PRP tem vários ambientes de aplicação, com destaque para *job shop* e *flow shop*, indo de variantes como flexível, sem espera, *buffers* limitados, entre outros.

Entre as estruturas de cromossomo, a mais comum é a baseada em operação seguida da baseada em tarefa.

Entre as estruturas de vizinhança a Representação Natural e o Caminho Crítico dominam.

Tabela 2.5 Resumo das características das abordagens da literatura e da tese (BACO).

Artigo	Problema	PP/PRP	EV	Métodos	Operadores vizinhança	Cenário	Validação
Roshanaei et al. (2009)	Job shop	PRP	RN	VNS	Insert	Taillard	Anova
Jarboni, Eddaly e Siarry (2011)	Flow shop	PP	RN	AG/VNS	Insert, Swap	Taillard, Reeves, Heller	ER
Zobolas, Tarantilis e Ioannou (2009)	Flow shop	PP	RN	AG/VNS	Insert, Swap, 2-opt	Taillard	NC
Song e Xu (2010)	Job shop	PP	RN	AG/CLS	Insert	Internet	NC
Sun et al. (2010)	Flow shop	PP	RN	PSO/VNS	Insert	Taillard	ER
Pan et al. (2011b)	Flow shop	PP	RN	DDEA/VNS	Insert, Swap	Taillard	ER
Yazdani, Amiri e Zandieh (2010)	Job shop	PP	RN	VNS	Swap	Vários	ER
Zheng e Yamashiro (2010)	Flow shop	PP	RN	QDEA/VNS	Insert, Swap, 2-opt	Taillard	ER
Tang et al. (2011)	Job shop	PP	CC	AG/PSO	Insert	Kacem	DR
Sels, Craeymeersch e Vanhoucke (2011)	Job shop	PP	CC	AG/VNS SS/VNS	Insert. Swap	Taillard	ER Médio
Wang e Zhang (2011)	Job shop	PP	CC	AM/VNS	Insert, Swap	OR-library	Médias
Vela, Varela e González (2010)	Job shop	PP	CC	AG/VNS	Insert, Swap, 2-opt	Várias	ER
Gao et al. (2011)	Job shop	PP	CC	AM/VNS	Insert, Swap	OR.library	
BACO	Job shop	PRP	AC	AG/VNS	Insert, Swap, 2-opt	Taillard	Teste de hipóteses

No AG, as taxas de cruzamento variam de 70% a 100%, sendo comum encontrar as operações PPX, OX, CX e PMX; e as taxas de mutação variam de 1% a 80% com operações *Insert*, *Swap* e 2-opt sendo muito usadas.

No VNS os operadores de vizinhança *Insert*, *Swap* e 2-opt são uma constante.

Nas validações o uso de bases da Internet são rotineiras, sendo os cenários vindos de Taillard e OR-library os mais usados.

Interessante notar que quando se fala na validação, o mais comum é que esta ocorra pela comparação dos resultados, diretamente ou por meio de seus erros relativos, sendo o uso de ferramentas estatísticas pouco usual.

O uso de hibridizações entre busca global e busca local já é uma constante nas pesquisas, sendo comum o uso de AG, PSO e ACO como buscas globais e VNS, SA e Busca Tabu como métodos de busca local.

2.4 Considerações finais

Neste capítulo foram apresentados os conceitos de representação de problema PRP, estruturas de vizinhança e operadores de vizinhança, AG e VNS, necessários para o entendimento desta tese.

No tocante a revisão da literatura levantam-se dados que sinalizam para o uso de AG e VNS, com uma estrutura mais correlata, além de vários parâmetros.

Tendo em vista as estruturas de vizinhança encontradas, que possuem baixa correlação com o *makespan* ou simplificam demasiadamente a estrutura, torna-se evidente a possibilidade de que "se for criada uma estrutura de vizinhança mais correlata ao *makespan* pode-se conseguir melhores *makespan* em um tempo de execução limitado".

Portanto, a proposta é apresentar uma nova estrutura de vizinhança aplicada em uma colaboração de busca global com busca local, objetivando minimizar o *makespan* em problemas de PRP. A validação se dá pela análise estatística dos resultados na execução desta proposta e de outros três trabalhos para comparação.

Diante da análise feita e do objetivo da proposta, esta validação se dá pela implementação e comparação dos resultados obtidos com os seguintes parâmetros:

- Método de busca global – Algoritmo Genético;
- Método de busca local – VNS;
- Problema a ser tratado - Programação Reativa da Produção;

- Ambiente de produção - *job shop*;
- Tipo do cromossomo – baseado em operações;
- Método de geração de população inicial - aleatório;
- Cruzamento – do tipo PPX com taxa de 66%;
- Mutação – usando operador 2-opt e taxa de 10%;
- Estruturas de vizinhança - Caminho Crítico, Representação Natural e a estrutura objeto desta proposta;
- Operadores de vizinhança - *Insert*, *Swap* e 2-Opt;
- Cenários -vindos de (TAILLARD, 2015).

Capítulo 3

BACO - BUSCA EM ÁRVORE DE COBERTURA

3.1 Considerações iniciais

A hipótese desta tese é a de que uma estrutura de vizinhança mais correlata à programação permite alcançar um *makespan* menor aos existentes para problemas de Programação Reativa da Produção.

A proposta é o uso desta estrutura de vizinhança aplicada em uma colaboração de AG e VNS.

Neste capítulo, esta estrutura de vizinhança é descrita e instanciada para melhor entendimento. Completando a descrição da proposta, são apresentados os pseudocódigos do VNS e da colaboração AG-VNS assim como seus parâmetros.

Todos os conceitos relevantes ao entendimento deste capítulo foram apresentados no Capítulo 2.

3.2 Proposta

Conforme levantamento realizado na fase de revisão da literatura, constatou-se que a busca por minimização de *makespan* em problemas de Programação Reativa da Produção tem sua relevância e atualidade e que o uso de hibridização ou colaboração de método global com

método local, inclusive do método de busca global AG com o método de busca local VNS, são bem referenciados.

Outra constatação, em se tratando de problemas de Programação Reativa da Produção, foi a de que as estruturas de vizinhanças usadas nas buscas locais possuem pouca correlação com a programação ou, quando possuem tal correlação, são de obtenção complexa ou simplificam demasiadamente o problema. Entretanto, para se conseguir uma boa resposta em uma busca local, a vizinhança deve estar correlacionada ao objetivo (AARST e LENSTRA, 2003).

A proposta é inédita ao criar uma nova estrutura de vizinhança baseada em árvore de cobertura (AC) para buscas que objetivam a minimização de *makespan*.

Em resumo, a proposta é a criação de uma estrutura de vizinhança baseada em árvore de cobertura altamente correlacionada com a programação, aplicada na busca local dentro de uma colaboração de AG com VNS com o objetivo de minimizar o *makespan* em Programação Reativa da Produção.

Nas próximas seções, estão descritas a colaboração entre o Algoritmo Genético e VNS, as escolhas de configurações usadas, e a proposta de estrutura de vizinhança.

3.2.1 **Descrição da Colaboração dos métodos de busca global e local**

Esta proposta busca verificar a qualidade do uso de uma nova estrutura de vizinhança baseada em árvore de cobertura, aplicada em uma colaboração dos métodos de busca AG e VNS. Nesta seção explica-se, detalhadamente, como esta colaboração acontece.

No Capítulo 2 o esquema genérico do AG e do VNS assim como seus algoritmos estão explicados. Aqui apresenta-se os parâmetros e configurações usados assim como o passo a passo da colaboração. Na próxima seção é apresentado a extração da estrutura BACO (Busca em Árvore do Cobertura) da solução e sua reinserção após ter sido transformado em um novo vizinho por algum operador de vizinhança.

Nesta colaboração são usadas as seguintes configurações para o AG:

- População de 30 indivíduos;

- Cromossomo como matriz linha, com número de genes igual ao número de operações em uma programação completa, sendo cada gene uma sugestão da tarefa a ser feita. Cada cromossomo representa uma sequência sugerida de tarefas para se produzir uma PRP;
- Geração de nova população:
    - Seleção por Elitismo: 1 indivíduo, selecionando-se aquele que possui o menor *makespan*;
    - Seleção por Roleta: 9 indivíduos;
    - População `Intermediária` de 20 indivíduos, formada pelas operações de cruzamento e mutação descritas a seguir:
        - Cruzamento: formam-se 2 grupos. O primeiro (`Grupo 1`) com os 10 melhores cromossomos da população desta geração e o segundo (`Grupo 2`) com o restante da população, que não pertence ao primeiro grupo. Gera-se, desta forma, 20 novos indivíduos com suas primeiras metades vindas do `Grupo 1` e suas segundas metades vindas do `Grupo 2`. Os cruzamentos seguem o método PPX, explicado na Subseção 2.2.2.1. Em seguida calculam-se os *makespans*;
        - Mutação: aplica-se operação 2-*opt* em 3 indivíduos, selecionados aleatoriamente entre os indivíduos gerados pela cruzamento;
    - Nova população: ao final das mutações, unem-se os indivíduos selecionados pelo mecanismo de elitismo e de roleta aos indivíduos da população `Intermediária`, sendo criada uma nova população de 30 indivíduos, que é passada para a próxima geração do AG.

Nesta colaboração são usadas as seguintes configurações para o VNS:

- Entrada: uma solução na forma de cromossomo, composta de `n x m` genes, cada um representando uma sugestão de tarefa a ser executada na PRP, onde `n` é o número de tarefas e `m` o número de máquinas do cenário;
- Operadores: *Insert*, *Swap* e *2-opt* todos operando com posições aleatórias;
- Forma de execução: operar 10 vezes com os operadores *Insert*, *Swap* e *2-opt*, intercalando-os;

- Saída: uma solução na forma de cromossomo, representando a nova solução obtida pela busca local.

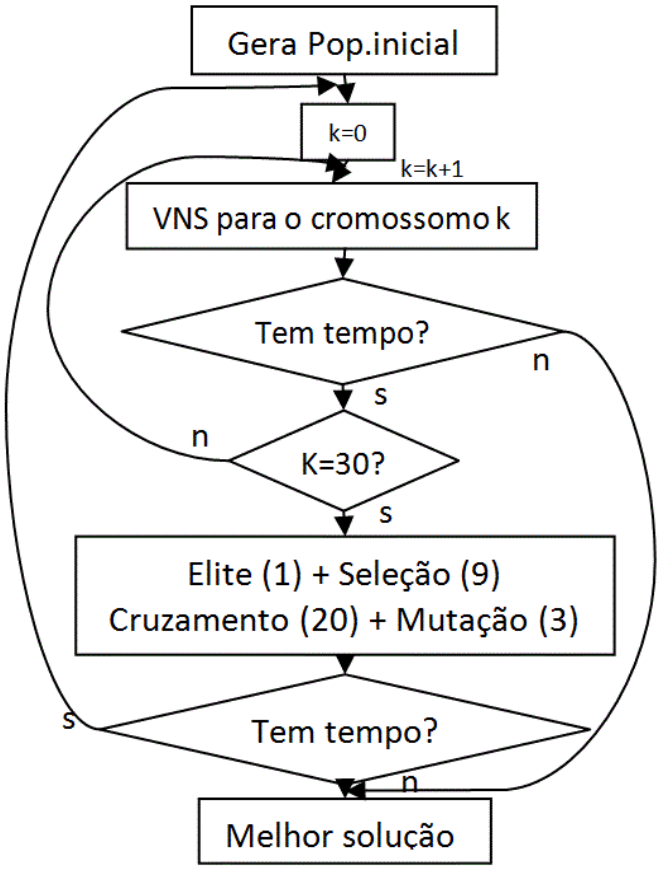

Figura 3.1 Fluxograma da colaboração entre AG e VNS.

Na Figura 3.1 pode-se ver o diagrama da colaboração entre o AG e o VNS. Abaixo, está o passo a passo elucidando como se dá a colaboração entre o AG e o VNS:

Passo 1 - O AG cria uma população Inicial com 30 cromossomos gerados aleatoriamente, formados por genes que indicam uma sugestão de sequência da programação;

Passo 2 – O VNS recebe todos os indivíduos da população Inicial, um a um, operando a vizinhança de cada indivíduo, trocando os indivíduos originais pelas suas versões melhoradas, quando ocorrerem.

Após cada busca local em um indivíduo, verificar se o tempo limite foi excedido, e, se tiver sido, salva-se as informações da corrente PRP gerada e termina-se a busca local e global, indo ao Passo 9.

O passo a passo do VNS é o seguinte:

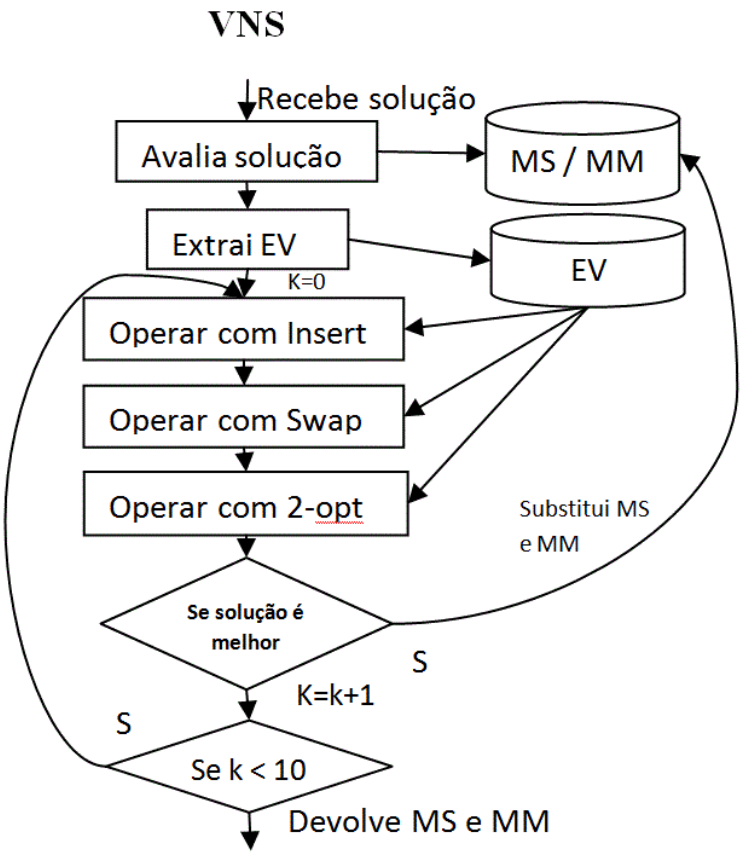

Figura 3.2 Diagrama do VNS na colaboração.

Passo 2.1 – Receber uma solução (na forma de cromossomo);

Passo 2.2 – Avaliar o *makespan* da solução (usando o processo indicado no Passo 2 da geração de estrutura da vizinhança, Seção 3.2.2) e salvar o resultado como `melhor_makespan` e a solução como `melhor_solução`;

Passo 2.3 – Extrair a vizinhança da solução, conforme explicado na Seção 3.2.2;

Passo 2.4 – `contador` recebe 0;

Passo 2.5 – Operar a vizinhança com operador *Insert*, retornar a vizinhança ao estado de solução como explicado na Seção 3.2.2, avaliar o *makespan*, comparar com `melhor_makespan` e, se for o melhor, salvar o novo *makespan* em `melhor_makespan` e salvar a solução em `melhor_solução`;

Passo 2.6 – Repetir o Passo 5 para o operador *Swap*;

Passo 2.7 – Repetir o Passo 5 para o operador *2-opt*;

Passo 2.8 – Incrementar o `contador` e, se for menor que 10, voltar ao Passo 5;

Passo 2.9 – Retornar `melhor_solução` na forma de cromossomo.

Passo 3 – Selecionar a elite (1 indivíduo) e colocar na população chamada `Nova`;

Passo 4 - Selecionar 9 indivíduos por roleta e colocar na população chamada `Nova`;

Passo 5 – Selecionar os 10 melhores indivíduos e colocá-los em uma população chamada `Melhores`. Em seguida, colocar os outros 20 indivíduos em uma população chamada `Piores`. Gerar 20 novos indivíduos, escolhendo aleatoriamente 1 indivíduo de cada população (`Melhores` e `Piores`) e cruzá-los usando o operador de cruzamento PPX. Colocar estes novos indivíduos na população `Intermediária`;

Passo 6 – Executar 3 mutações na população `Intermediária`, definida pelo uso do operador 2-*opt*;

Passo 7 – Esvaziar a população `Inicial` e transferir todos os indivíduos das populações `Nova` e `Intermediária` para a população `Inicial`;

Passo 8 – Se a condição de parada não foi alcançada (que é o tempo limite de cinco minutos) retorna-se ao Passo 2. Caso contrário, segue para o Passo 9;

Passo 9 - Termina-se o processamento e retorna-se ao melhor cromossomo obtido.

3.2.2 **Descrição da Estrutura de vizinhança BACO**

Na revisão realizada, o cromossomo geralmente é uma lista de genes que representam as operações da programação das operações, conhecida como representação de cromossomo baseada em operações (CHENG, 1996), como apresentado no Capítulo 2. Cada gene representa uma sugestão de operação, normalmente indicando a tarefa da operação.

O uso de Redes de Petri e árvore de cobertura como representação de problemas de PRP e suas soluções possui alta correlação com a programação, quando o objetivo é minimizar o *makespan*. Permite, ainda, uma boa visualização gráfica e é adequada a problemas com compartilhamento de recursos. Um problema desta forma representado torna possível sua manipulação matemática com o uso das equações de estados e matriz de transição.

Deste modo, uma estrutura baseada nesta árvore de cobertura permite uma rápida manipulação de suas soluções, tornando boas soluções em soluções vizinhas promissoras.

Nesta proposta, o cromossomo inicialmente é uma sugestão de tarefas. Durante a geração da programação equivalente ao cromossomo mapeia-se esta solução na árvore de cobertura associada.

O caminho nesta árvore de cobertura é transposto em uma lista ordenada pelo tempo de início das operações, na qual cada elemento acumula todas as operações que possuem o mesmo tempo de início. Esta lista possui um número de elementos menor que o número de tuplas do caminho. Esta nova lista é a estrutura de vizinhança proposta.

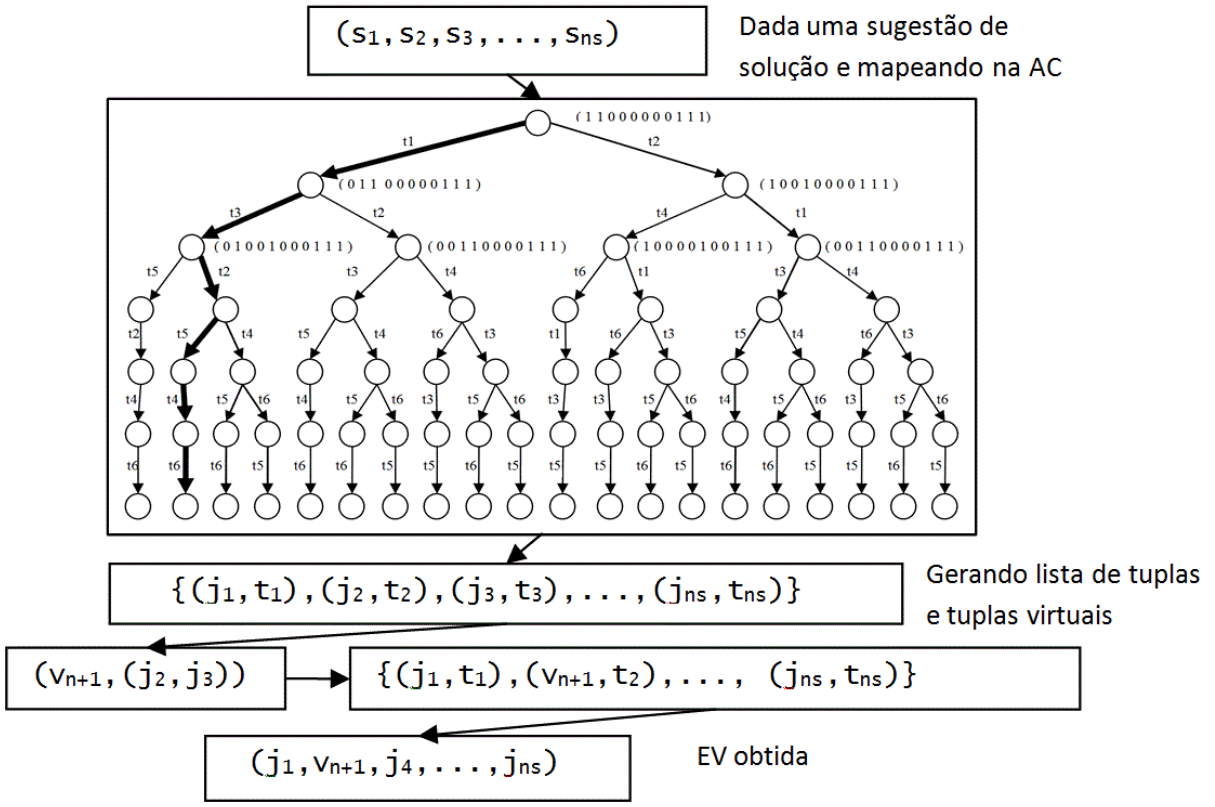

Figura 3.3 Fluxograma de como se dá a extração da estrutura de vizinhança da proposta.

A Figura 3.3 descreve como se dá a extração da vizinhança de uma solução dada como entrada no VNS. Esta extração é o cerne da hipótese desta tese. Um passo a passo sobre o processo e figuras explicativas se encontra na próxima seção.

Considere como entrada o conjunto ordenado de sugestões de tarefas S, com ns sugestões de operações, a tupla (j_i, t_i) como parte do mapeamento na AC sendo j_i a tarefa e t_i o tempo de início da operação, e n como sendo o número de tarefas da programação a ser obtida.

Resumindo:

- A solução é transposta na AC;
- Em seguida, gera-se uma lista de tuplas com a sequência em que as transições são disparadas, listando apenas as tarefas associadas à transição e o seu tempo de início;
- Buscam-se, nesta lista, todas as tuplas que possuem o mesmo tempo de início e as substitui por tuplas virtuais;
- Cria-se uma lista ordenada com as tarefas da lista anterior. Esta lista é a estrutura de vizinhança da proposta.

A descrição passo a passo da criação da estrutura de vizinhança proposta é a seguinte:

Passo 1 – Receber uma sugestão de programação na forma de cromossomo, visto que a colaboração acontece com um AG. Esta sugestão de solução é uma lista de tarefas, sendo cada tarefa repetida um número de vezes igual ao número de máquinas.

Genericamente seja $S=\{s_1, s_2, s_3, ... s_{ns}\}$ onde s_i é uma sugestão de tarefa e i varia de 1 até o número de operações da programação ($ns = n \times m$). Cada sugestão de uma tarefa específica aparece um número de vezes igual ao número de máquinas constante em seu roteiro.

Passo 2 – Mapear o cromossomo de sugestões de programação de tarefas na AC, de acordo com as regras de atribuição de operações definidas.

Este mapeamento ocorre por meio da execução da Rede de Petri associada ao problema (cenário), utilizando o cromossomo de sugestões de sequência de tarefas como guia e solucionador de conflitos. Conforme as transições vão sendo disparadas, vai-se mapeando na AC. Neste momento, também são geradas as tuplas da programação e estas tuplas são da forma (transição disparada, a tarefa sendo executada, a máquina em uso, o tempo de início da operação, o tempo final da operação) e após sua geração são adicionadas à lista de operações da programação em curso. Cada transição disparada implica em se retirar da lista de sugestões a tarefa correspondente mais à esquerda.

A todo momento, na RP, são possíveis 4 situações, e, ao final do tratamento de cada situação (com sua sequência de ações), retorna a analisar a próxima situação na RP. Esta análise se repete até que seja executada a primeira situação, que indica o final da execução. As situações são as seguintes:

1) Nenhuma transição está habilitada e não há transições sendo executadas (a execução da RP terminou e o tempo da última transição finalizada é igual ao *makespan* desta programação);

2) Nenhuma transição está habilitada e há pelo menos uma transição em execução (avança-se para o término da transição com final mais próximo);
3) Apenas uma transição está habilitada (dispara-se esta transição, mapeia-se na AC, exclui-se a primeira aparição desta tarefa na lista de sugestões, cria-se tupla com dados da operação iniciada e coloca-se esta tupla na lista da programação);
4) Há mais de uma transição habilitada (escolhe-se a transição que representa a tarefa que está mais próxima ao início da lista de sugestões. Dispara-se esta transição, mapeia-se na AC, exclui-se a primeira aparição desta tarefa na lista de sugestões, cria-se tupla com dados da operação iniciada e coloca-se esta tupla na lista da programação).

Passo 3 – Copiar a sequência de transições do caminho na AC, produzindo uma lista ordenada das tuplas das transições disparadas. Cada tupla faz referência à tarefa a qual pertence a operação a ser executada e o tempo de início de tal operação. Esta lista é ordenada pelo tempo de início.

Passo 4 – Reduzir a quantidade de tuplas da sequência gerada no Passo 3 por meio da criação de tuplas virtuais. As tuplas virtuais são tuplas que, acumulam sob um código de tarefa virtual, mais de uma tupla real, possuindo o mesmo tempo de início das tuplas reais. Os códigos de tarefas virtuais iniciam-se em (número de tarefas + 1) e são incrementados a cada nova criação. Este processo ocorre pela identificação de todos os blocos de tuplas que possuem o mesmo tempo de início, sendo os blocos retirados da sequência e as tuplas virtuais inseridas na sequência.

Após as substituições possui-se uma sequência de tuplas (reais e virtuais) com número menor de tuplas e uma tabela relacionando os códigos de tarefas virtuais às tarefas das operações substituídas.

Seja (j_i, t_i) uma tupla genérica e seja $0 < i <= ns$, onde ns é o número total de operações na programação. Nestas tuplas j_i é a referência à

tarefa a quem pertence esta operação e t_i é a referência ao tempo de início desta operação. Caso exista mais de uma tupla em que $t_i = t_j$, com $i <> j$, cria-se uma tupla virtual (v_k, t_i) com v_k sendo um código atribuído de forma única a esta tupla e eliminam-se da sequência original todas as tuplas que continham um tempo de início igual a t_i. Adiciona-se ao conjunto de relacionamentos de tuplas virtuais o código v_k e os j_i substituídos.

Genericamente, considere a sequência $((j_1, t_1) (j_2, t_2) (j_3, t_3))$ e o número de tarefas n, e que os tempos t_2 e t_3 sejam iguais, então se cria um código de tarefa virtual v_k, com v_k igual a $n + 1$, e a tupla virtual (v_k, t_2). Excluem-se da sequência as tuplas que possuem tempo de início igual a t_2 e insere-se a tupla virtual, obtendo-se ao final a sequência $((j_1,t_1) (v_k, t_2))$ e uma tupla relacionando o código de tarefa virtual v_k às tarefas j_2 e j_3 $(v_k, (j_2,j_3)$. Esta tupla de relacionamento é colocada no conjunto de relacionamentos virtuais $\{(v_k,(j_2,j_3)\}$.

Passo 5 – Criar uma lista de tarefas codificadas baseada na sequência criada no passo 4. Para criá-la, inicia-se com uma lista vazia e, para cada tupla da sequência, retira-se o elemento que faz referência à tarefa real ou virtual e insere–se tal elemento na lista, a qual é a nova estrutura de vizinhança.

Seguindo a instanciação ou exemplo indicado no Passo 4, da sequência (j_1,t_1) (v_k, t_2) extraem-se os elementos j_1 e v_k e cria-se a lista (j_1, v_k).

Passo 6 – Retorna-se esta lista como a estrutura de vizinhança e o conjunto de relacionamentos virtuais. No genérico (j_1,v_k) e $\{(v_k,(j_2,j_3)\}$ respectivamente.

O algoritmo de retorno da estrutura de vizinhança para o cromossomo é o seguinte:

Passo 1 – Receber a vizinhança e o conjunto de relacionamentos virtuais. A vizinhança está na forma indicada no Passo 5 e o conjunto de relacionamentos virtuais é o indicado no final do Passo 4. Pelo exemplo

genérico apresentado nos passos 4 e 5 do algoritmo anterior são respectivamente (j_1, v_k) e $\{(v_k, (j_2,j_3)\}$.

Passo 2 – Substituir na lista, que é a vizinhança, os códigos de tarefas virtuais por códigos de tarefas reais. Cada tupla virtual é desdobrada em um número de códigos reais igual ao número de relacionamentos indicados no conjunto de relacionamentos virtuais.

Genericamente, dados a lista (j_1, v_k) e o conjunto de relacionamentos $\{(v_k, (j_2,j_3)\}$ desdobra-se o código virtual v_k nos códigos reais j_2 e j_3, e substitui-se o código virtual v_k na lista pelos códigos reais j_2 e j_3 obtendo-se a lista (j_1, j_2, j_3).

Passo 3 – Substituir o cromossomo pela sequência de tarefas indicada na lista, o que, genericamente, é equivalente a retornar como cromossomo (para ser retrabalhado pelo AG) a lista de tarefas (j_1, j_2, j_3), ficando o novo cromossomo como (j_1, j_2, j_3).

Exemplificando:

Considere o problema apresentado no Capítulo 2, na seção Rede de Petri, de 2 tarefas por 3 máquinas mostrado na Figura 2.8. Esta figura está repetida na Figura 3.4 para facilitar o acompanhamento. Nela, verifica-se 6 transições e 11 lugares, sendo que 3 destes lugares representam as máquinas (ou recursos compartilhados).

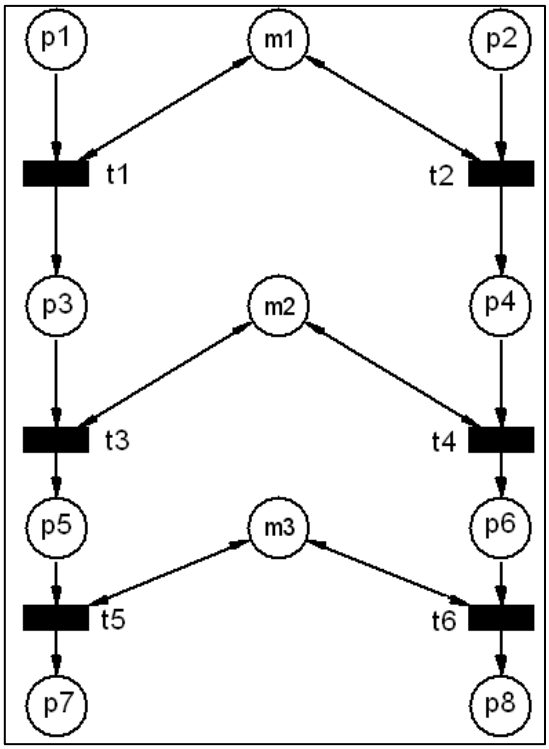

Figura 3.4 Rede de Petri do exemplo de cenário simples.(Fonte: autor)

Os lugares `p1, p3, p5, p7` e `p2, p4, p6, p8` são as etapas de processamento (operações) das duas tarefas; as transições `t1, t3, t5 e t2, t4, t6` são os roteiros destas tarefas; e `m1, m2, m3` são os recursos sendo compartilhados. Percebe-se que ambas as tarefas disputam os recursos `m1, m2 e m3`.

Para esta Rede de Petri, a árvore de cobertura correspondente é dada na Figura 3.5, onde se observam os nós representando cada conjunto de marcações e as setas indicando que a transição leva de um conjunto de marcação para outro.

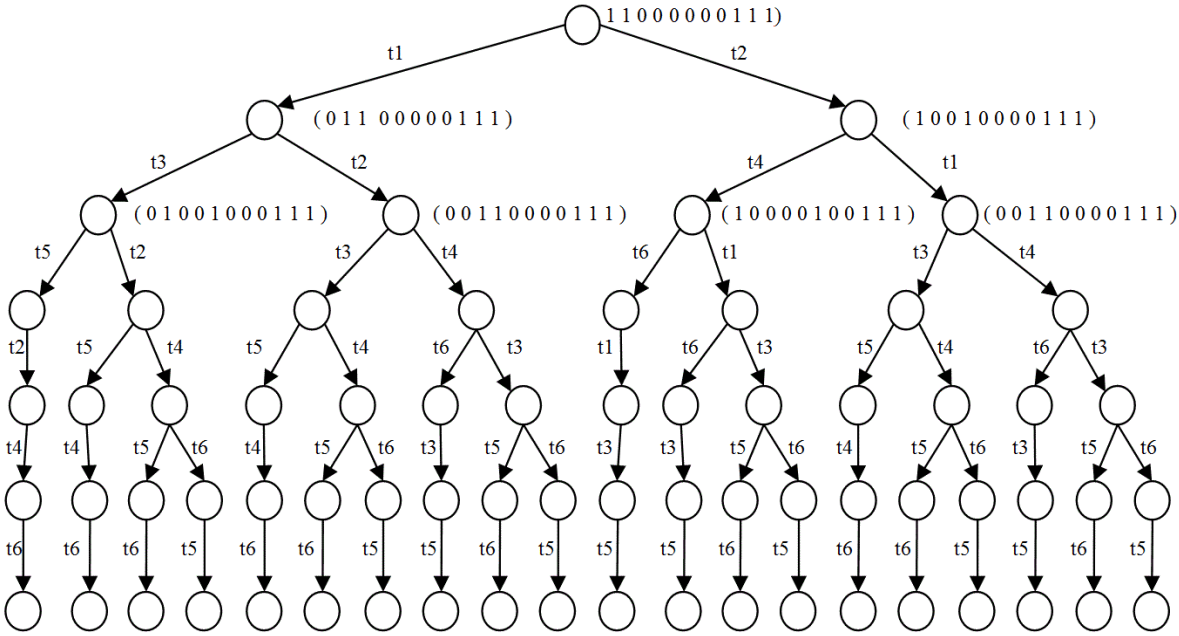

Figura 3.5 Árvore de cobertura da Rede de Petri da Figura 3.4.

Considere este problema sendo um *job shop* de duas tarefas com 3 máquinas, representado na forma de 2 tabelas: a Tabela 3.1 contendo os roteiros e a Tabela 3.2 contendo os tempos de produção em cada fase.

Tabela 3.1 Roteiros para um conjunto de tarefas do exemplo, indicando as máquinas de cada operação.

	Primeira operação	Segunda operação	Terceira operação
Tarefa 1	m1	m2	m3
Tarefa 2	m1	m2	m3

Tabela 3.2 Tempos de operação de cada fase de um conjunto de tarefas do exemplo.

	Tempo operação 1	Tempo operação 2	Tempo operação 3
Tarefa 1	30 ut	9 ut	5 ut
Tarefa 2	10 ut	10 ut	20 ut

Executando o algoritmo de extração da vizinhança, tomando um cromossomo de entrada:

Passo 1 – Receber o cromossomo contendo uma sugestão de programação, sendo cada gene uma sugestão de tarefa:

1	1	1	2	2	2

Para o processo de execução na RP, este cromossomo é transformado em uma lista de sugestões (1,1,1,2,2,2).

Passo 2 – Mapear o cromossomo, de acordo com as regras de atribuição de operações definidas na AC.

Este mapeamento ocorre por meio da execução da RP da Figura 3.4, utilizando o cromossomo de sugestões de tarefas como guia e solucionador de conflitos. Conforme as transições vão sendo disparadas, vai-se mapeando na AC. Neste momento, também são geradas as tuplas da programação e estas tuplas são da forma (transição disparada, a tarefa sendo executada, a máquina em uso, o tempo de início da operação, o tempo final da operação) e, após sua geração, são adicionadas à lista de operações da PRP em curso. Note que esta tupla é parecida mas não igual à tupla definida no Capítulo 2, onde definiu-se a solução genérica.

Todo o processo da execução na RP está ilustrado nas figuras 3.6, 3.7 e 3.8, que mostram a RP do problema simplificada, sem os nomes das transições e lugares. Embora o mapeamento na AC ocorra a cada transição disparada, optou-se por mostrar somente a situação da AC ao final da execução, na Figura 3.9.

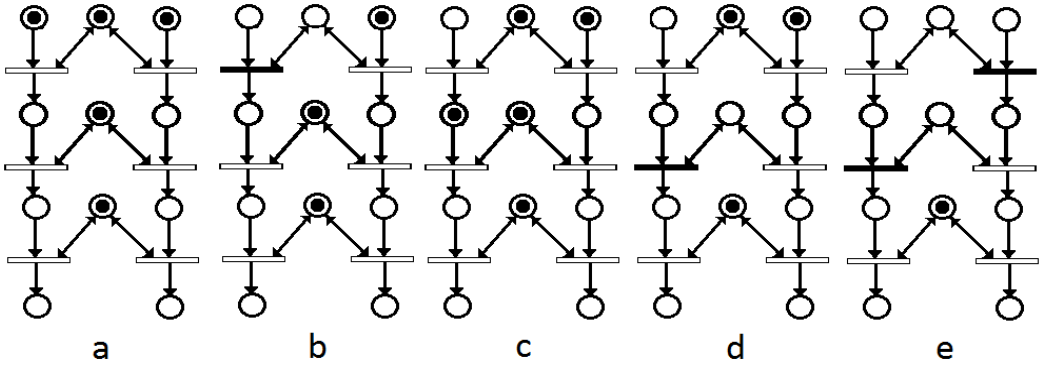

Figura 3.6 Fases da execução do exemplo na Rede de Petri (a - situação inicial, b - ativação da transição t1, c - final da transição t1, d – ativação da transição t3, e - ativação da transição t2).

A seguir, descreve-se todo processo de execução do cromossomo na RP, explicando-se as habilitações e disparos de transições, gerenciamento da lista de sugestões e criação de tuplas de programação assim como dos mapeamentos na AC.

A Rede de Petri (RP) na Figura 3.6a representa a situação inicial e tem marcações em todas as máquinas (m1, m2 e m3) e no início de cada tarefa (p1 e p2). As transições t1 e t2 estão habilitadas e deve-se resolver qual das duas deve ser disparada.

Resolve-se este conflito olhando a lista de sugestões (1,1,1,2,2,2), na qual a primeira sugestão indica a tarefa 1 a ser executada, em seguida, dispara-se a transição t1 (Figura 3.6b), cria-se a tupla (t1, 1, m1, 0 , 30), mapeia-se na AC e retira-se a 1ª sugestão da lista (1,1,2,2,2).

Após este disparo (Figura 3.6b), a RP não possui nenhuma transição habilitada. Avança-se o tempo até o término da transição t1 (Figura 3.6c), o que ocorre no tempo 30 ut, e, neste ponto, tem-se duas transições habilitadas, t3 e t2.

Seguindo a lista de sugestões (1,1,2,2,2), a 1ª sugestão é a tarefa 1, que indica a transição t3. Dispara-se esta transição (Figura 3.6d), cria-se a tupla (t3, 1, m2, 30, 39), mapeia-se na AC e retira-se a sugestão da lista (1,2,2,2).

Na Figura 3.6d, ainda nota-se que somente a transição t2 está habilitada e, na lista de sugestões (1,2,2,2), só a 2ª sugestão. Dispara-se a transição t2 (Figura 3.6.e), retira-se da lista a 2ª sugestão (1,2,2) e cria-se a tupla (t2, 2, m1, 30, 40).

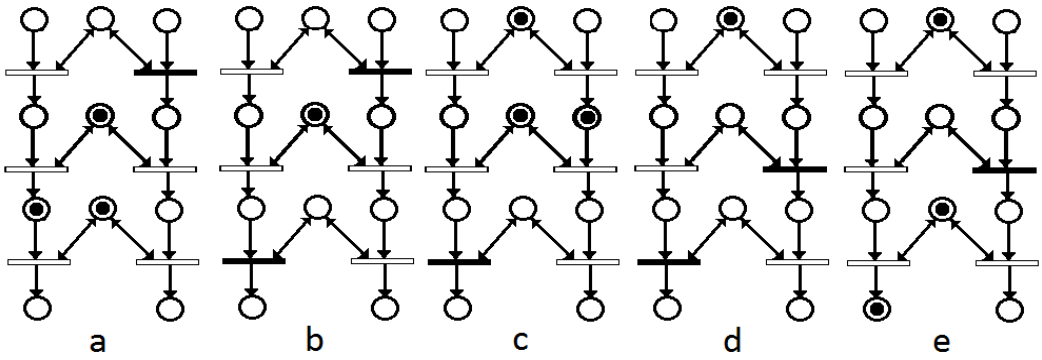

Figura 3.7 Fases da execução do exemplo na Rede de Petri (a – final da transição t3, b - ativação da transição t5, c - final da transição t2, d – ativação da transição t4, e – final da transição t5).

Novamente, não há transições habilitadas na Figura 3.6e e, portanto, avança-se o tempo para o término da transição com tempo final menor (no caso, a transição t3 em 39 ut).

Na situação mostrada na Figura 3.7a, somente a transição t5 está habilitada. Então, dispara-se a transição t5 (Figura 3.7b), busca-se a próxima sugestão com a tarefa 1 na lista de sugestões (1,2,2), exclui-se (2,2) e cria-se a tupla da operação (t5, 1, m3, 39, 44).

Sem transições habilitadas, novamente, avança-se para o tempo de término da transição com final mais próximo, no caso, a transição t2 no tempo 40 ut (Figura 3.7c). Passa-se a ter a transição t4 habilitada, e somente ela. Dispara-se a t4 (Figura 3.7d), retira-se da lista de sugestões (2,2) a 1ª ocorrência da tarefa 2, que é a responsável pela transição t4, ficando a lista de sugestões com apenas uma tarefa (2), e cria-se a tupla de operação (t4, 2, m2, 40, 50).

Novamente, não há transições habilitadas, então se avança para o término da transição com final mais próximo, no caso, a transição t5 no tempo 44 ut (Figura 3.7e).

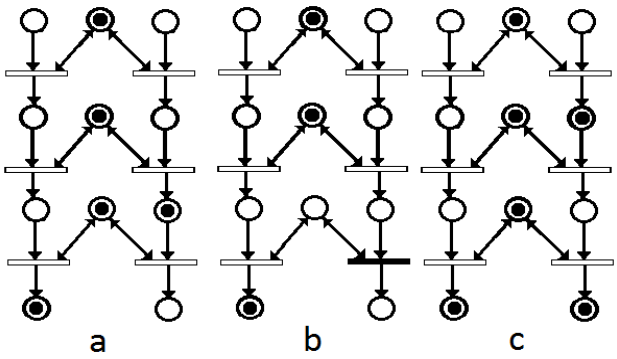

Figura 3.8 Fases da execução do exemplo na Rede de Petri (a – final da transição t4, b - ativação da transição t6, c - final da transição t6 e situação final da Rede de Petri).

Na Figura 3.7e, continua-se sem alguma transição habilitada. Busca-se a próxima transição com término mais próximo, no caso, a transição t4 no tempo 50 ut, e a finaliza (Figura 3.8a).

Na Figura 3.8a, somente a transição t6 está habilitada. Busca-se na lista a tarefa 2, a última e única da lista de sugestões, retira-se da lista, cria-se a tupla (t6, 2, m3, 50, 70) e dispara-se a transição t6 (Figura 3.8b).

Na Figura 3.8b, não há mais transições habilitadas. Avança-se para o término da transição com final mais próximo, no caso, t6 no tempo de 70 ut. Finaliza-se a transição e a execução da sugestão de programação, obtendo-se o *makespan* desta programação, que é de 70 ut.

Seguindo este exemplo, produziu-se uma lista de tuplas que indicam a programação, referenciando transições, tarefas, máquinas, tempos de início e fim de cada operação.

Esta lista é a seguinte: ((t1, 1, m1, 0, 30), (t3, 1, m2, 30, 39), (t2, 2, m1, 30, 40), (t5, 1, m3, 39, 44), (t4, 2, m2, 40, 50), (t6, 2, m3, 50, 70)).

A Figura 3.9, mostra o mapeamento desta sequência de ativações (t1, t3, t2, t5, t4, t6) na AC. O caminho feito na árvore de cobertura corresponde exatamente à sequência de ativações das transições para a programação executada.

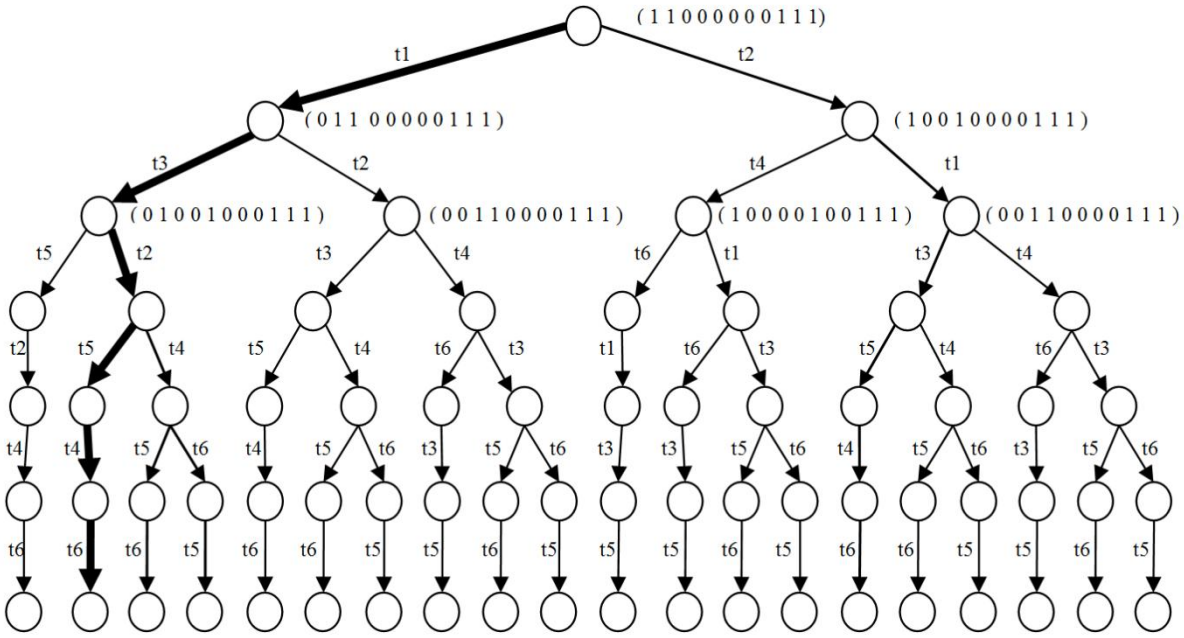

Figura 3.9 Caminho do cromossomo sugestão na AC.

Passo 3 – Copiar a sequência de transições do caminho na AC, criando uma lista de tuplas com a tarefa e o tempo de início:

Extrai-se da lista de tuplas de programação uma lista equivalente, mas sem conter as referências à transição, máquina e tempo final. A nova lista é ((1, 0), (1, 30), (2, 30), (1, 39), (2, 40), (2, 50)).

Passo 4 – Reduzir a quantidade de tuplas da sequência gerada no Passo 3 por meio da criação de tuplas virtuais. As tuplas virtuais são tuplas que acumulam, sob um código de tarefa virtual, mais de uma tupla real, possuindo o mesmo tempo de início das tuplas reais. Este processo ocorre pela identificação de todos os blocos de tuplas, que possuem o mesmo tempo de início, sendo os blocos retirados da sequência e as tuplas virtuais inseridas na sequência.

Após as substituições, tem-se uma sequência de tuplas (reais e virtuais) com número menor de tuplas e uma tabela relacionando os códigos de tarefas virtuais às tarefas das operações substituídas.

A numeração dos códigos das tuplas virtuais começa a partir do próximo número após o número de tarefas reais.

Observa-se que as tuplas (1, 30) e (2, 30), vindas do Passo 3, possuem o mesmo tempo de início de operação (segundo elemento).

Cria-se um código para a tupla virtual. Neste exemplo, que possui apenas duas tarefas, a numeração dos códigos das tuplas virtuais começa no número 3. Cria-se a tupla virtual (3, 30).

Exclui-se as tuplas (1, 30) e (2, 30) da sequência vinda do Passo 3 e acrescenta-se a tupla virtual (3, 30), obtendo-se a sequência (1, 0), (3, 30), (1, 39), (2, 40), (2, 50). Gera-se uma entrada na tabela (conjunto) de codificação das tarefas virtuais com as tarefas reais obtendo-se {(3, (1,2))}.

Passo 5 – Transformar em lista de tarefas codificadas e guardar tabela de codificação:

Neste passo usa-se a lista de tuplas obtida no Passo 4, extraindo a tarefa de cada tupla (primeiro elemento) e transferindo–as para uma nova lista, que é a lista de tarefas codificadas.

Lista de tarefas codificadas = (1, 3, 1, 2, 2);

Passo 6 – Retorna-se esta lista como sendo a estrutura de vizinhança e o conjunto de relacionamentos virtuais.

Tabela de codificação = { (3, (1, 2))};

Nova estrutura de vizinhança = (1, 3, 1, 2, 2).

Exemplo de operação na vizinhança (usando o operador *swap*):

Considere que tal vizinhança tenha sofrido uma operação de *swap* entre as posições 1 e 4, ficando então

`1, 3, 1, 2, 2 → 2, 3, 1, 1, 2`

Exemplo de recodificação de vizinhança em novo cromossomo:

Passo 1 – Receber a vizinhança e a tabela de codificação:

Vizinhança= 2, 3, 1, 1, 2

Tabela de codificação = (tarefa 3 = tarefa 1 e tarefa 2);

Passo 2 – Substituir na vizinhança as codificações pelos seus significados:

Vizinhança= 2, 3, 1, 1, 2

Substituindo a tarefa 3 → 2, 1, 2, 1, 1, 2

Passo 3 – Substituir o cromossomo pela sequência de tarefas indicada na lista:

2	1	2	1	1	2

A Figura 3.10 mostra a AC com este cromossomo mapeado.

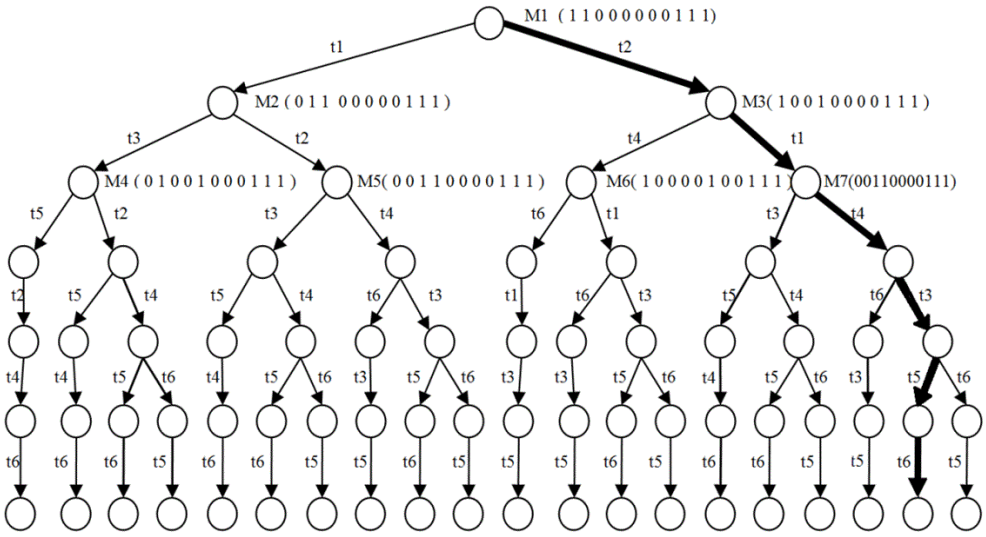

Figura 3.10 Árvore de cobertura para a Rede de Petri da Figura 3.3, mostrando a programação do cromossomo após operação na vizinhança.

3.3 Considerações finais

A hipótese desta tese de doutorado é a de que uma estrutura de vizinhança mais correlata à programação permite alcançar um *makespan* menor ao obtido pelos atuais métodos em uso para problemas de programação, com tempo de processamento relativamente curto e limitado (Programação Reativa da Produção). A proposta desta tese é a criação de uma estrutura de vizinhança baseada em árvore de cobertura, aplicada em uma colaboração de AG com VNS.

Neste capítulo, foi apresentada a proposta de estrutura de vizinhança baseada em AC aplicada na colaboração de AG com VNS, especificando-se os parâmetros e explicando-se todos os algoritmos pertinentes. A estrutura de vizinhança foi explicada de diferentes modos para garantir seu perfeito entendimento.

Pretende-se que, ao final da leitura deste capítulo, o leitor tenha compreendido como esta proposta está definida, sendo possível implementá-la se necessário.

Capítulo 4

VALIDAÇÃO

4.1 Considerações iniciais

A hipótese levantada nesta tese é: o uso de uma estrutura de vizinhança mais correlata à programação das operações aplicada em um sistema colaborativo de busca global-local obtém resultados melhores aos obtidos pelas estruturas de vizinhança atuais.

Este trabalho se pauta pelo método hipotético-dedutivo tendo em vista que permite corroborar uma hipótese dentro de um determinado contexto e contra determinados trabalhos alternativos. O resultado está sujeito ao momento em que ocorre, podendo no futuro, quando posto em comparação a outros concorrentes e/ou outros contextos, obter resultado diferente.

A validação é realizada pela comparação de quatro trabalhos: o desta proposta e outros três trabalhos que defendem outras técnicas. Para que a comparação tenha maior abrangência, cenários de diversos portes serão usados.

Neste capítulo, serão descritos o contexto da validação, o planejamento dos experimentos, seus resultados e as análises da execução dos experimentos planejados.

4.2 Contexto

Nesta validação, busca-se saber se a estrutura de vizinhança proposta, aplicada em uma colaboração de AG com VNS, dado um tempo limite, buscando o menor *makespan* para a PRP, é melhor do que as outras técnicas.

A validação se dará pela comparação dos resultados obtidos de forma empírica e a análise foi realizada usando ferramentas de estatística. Tornou-se necessário definir o contexto em que tais dados foram gerados assim como qual o ambiente computacional usado.

O contexto implica o problema tratado, os cenários (que são as entradas de dados) e os outros trabalhos a serem comparados. Tal contexto é importante porque se deve ter em mente a possibilidade de poder replicar o experimento em momento futuro. O ambiente computacional também é relevante, pois o problema tratado considera uma limitação de tempo e, para que uma replicação seja possível, é necessário levar em consideração o desempenho computacional disponível nesta validação.

Visando a validação, programou-se um sistema colaborativo AG-VNS com as características dadas no capítulo anterior. Esse sistema foi implementado e executado usando um processador i5 3337 com 4 GB de memória e o ambiente de programação Matlab 2009a.

4.2.1 **O Problema Tratado**

Na definição do contexto, primeiro estabeleceu-se o problema de Programação Reativa da Produção, sendo escolhidos cenários de *job shop*, por indicação da revisão de literatura.

Este problema é tratado com limitação de tempo em 300 segundos, para se obter as programações.

A ordem de complexidade deste problema é *np-hard*, tornando-se intratável na medida em que a dimensão do problema aumenta.

4.2.2 **Cenários**

Cenários são os conjuntos de dados de entrada para cada problema, contendo os roteiros de todas as tarefas, indicando máquina e tempo de cada operação.

Para que os experimentos tivessem maior abrangência, cenários descrevendo várias dimensões do problema foram utilizados.

Como segundo passo da definição do contexto da validação, determinou-se a origem dos dados (base de cenários) e quais cenários seriam utilizados, sendo escolhida a base de dados Taillard (TAILLARD, 1993; TAILLARD, 2015) por indicação da revisão de literatura, e, desta base, foram escolhidos três cenários de tamanhos diferentes, de forma a permitir maior abrangência do contexto. A escolha norteou-se por variar as dimensões, indo de 15 tarefas por 15 máquinas a 50 tarefas por 20 máquinas, e optou-se por selecionar sempre o primeiro cenário em cada categoria.

Os tempos limites para o processamento de todos os PRP vieram da experiência dos pesquisadores do Tear e foi definido com 300 segundos.

As tabelas 4.1 e 4.2 apresentam os dados referentes ao cenário de 15 tarefas por 15 máquinas.

Tabela 4.1 Roteiros de máquinas do cenário de *job shop* 15 tarefas x 15 máquinas(ta01).

	Máquina responsável pela etapa da tarefa														
Tarefa 1	7	13	5	8	4	3	11	12	9	15	10	14	6	1	2
Tarefa 2	5	6	8	15	14	9	12	10	7	11	1	4	13	2	3
Tarefa 3	2	9	10	13	7	12	14	6	1	3	8	11	5	4	15
Tarefa 4	6	3	10	7	11	1	14	5	8	15	12	9	13	2	4
Tarefa 5	8	9	7	11	5	10	3	15	13	6	2	14	12	1	4
Tarefa 6	6	4	13	14	12	5	15	8	3	2	11	1	10	7	9
Tarefa 7	13	4	8	9	15	7	2	12	5	6	3	11	1	14	10
Tarefa 8	12	6	1	8	13	14	15	2	3	9	5	4	10	7	11
Tarefa 9	11	12	7	15	1	2	3	6	13	5	9	8	10	14	4
Tarefa 10	7	12	10	3	9	1	14	4	11	8	2	13	15	5	6
Tarefa 11	5	8	14	1	6	13	7	9	15	11	4	2	12	10	3
Tarefa 12	3	15	1	13	7	11	8	6	9	10	14	2	4	12	5
Tarefa 13	6	9	11	3	4	7	10	1	14	5	2	12	13	8	15
Tarefa 14	9	15	5	14	6	7	10	2	13	8	12	11	4	3	1
Tarefa 15	11	9	13	7	5	2	14	15	12	1	8	4	3	10	6

Tabela 4.2 Tempos de operações do cenário de *job shop* 15 tarefas x 15 máquinas(ta01).

	Tempo de duração de cada etapa da tarefa														
Tarefa 1	94	66	10	53	26	15	65	82	10	27	93	92	96	70	83
Tarefa 2	74	31	88	51	57	78	8	7	91	79	18	51	18	99	33
Tarefa 3	4	82	40	86	50	54	21	6	54	68	82	20	39	35	68
Tarefa 4	73	23	30	30	53	94	58	93	32	91	30	56	27	92	9
Tarefa 5	78	23	21	60	36	29	95	99	79	76	93	42	52	42	96
Tarefa 6	29	61	88	70	16	31	65	83	78	26	50	87	62	14	30
Tarefa 7	18	75	20	4	91	68	19	54	85	73	43	24	37	87	66
Tarefa 8	32	52	9	49	61	35	99	62	6	62	7	80	3	57	7
Tarefa 9	85	30	96	91	13	87	82	83	78	56	85	8	66	88	15
Tarefa 10	5	59	30	60	41	17	66	89	78	88	69	45	82	6	13
Tarefa 11	90	27	1	8	91	80	89	49	32	28	90	93	6	35	73
Tarefa 12	47	43	75	8	51	3	84	34	28	60	69	45	67	58	87
Tarefa 13	65	62	97	20	31	33	33	77	50	80	48	90	75	96	44
Tarefa 14	28	21	51	75	17	89	59	56	63	18	17	30	16	7	35
Tarefa 15	57	16	42	34	37	26	68	73	5	8	12	87	83	20	97

As tabelas apresentando os outros cenários assim como o cenário 15x15 estão disponíveis no Apêndice A.

4.2.3 **Trabalhos usados na comparação com esta proposta**

A validação será realizada pela comparação de quatro trabalhos aplicados na solução do problema de Programação Reativa da Produção em *job shop* em três cenários.

Como a obtenção dos dados para comparação se deu por meio de experimentos, existiu a necessidade de se definir quais trabalhos seriam confrontados com a proposta da tese. A definição de quais trabalhos seriam confrontados é o terceiro passo na descrição do contexto. Estes trabalhos são oriundos da revisão de literatura, sendo os seguintes:

- Método de busca global puro [CHENG; GEN; TSUJIMURA, 1996; FALKENAUER; BOUFFOUIX, 1991; POTVIN, 1996];
- Método de busca global colaborativo com busca local usando estrutura de vizinhança Representação Natural; [JARBONI; EDDALY; SIARRY, 2011; ZOBOLAS; TARANTILIS; IOANNOU, 2009];
- Método de busca global colaborativo com busca local usando estrutura de vizinhança Caminho Crítico [SELS; CRAEYMEERSCH; VANHOUCKE, 2011; WANG; ZHANG, 2011].

As configurações e denominações para os quatro trabalhos estão descritas abaixo (as siglas indicam o trabalho (T) e a configuração de cada trabalho):

- **TBACO** – método de busca global (AG) colaborativa com método de busca local (VNS) usando como estrutura de vizinhança a estrutura proposta no Capítulo 3 (BACO);
- **TGLOBAL** – método de busca global (AG) sem o uso de busca local;
- **TRN** – método de busca global (AG) colaborativa com método de busca local (VNS) usando como estrutura de vizinhança a estrutura Representação Natural (RN);
- **TCC** – método de busca global (AG) colaborativa com método de busca local (VNS) usando como estrutura de vizinhança a estrutura Caminho Crítico (CC).

Observe que os trabalhos TBACO, TRN e TCC se distinguem apenas pela estrutura de vizinhança usada. O trabalho TGLOBAL não possui busca local nem estrutura de vizinhança.

Os parâmetros usados tanto no AG como no VNS, para os quatro trabalhos, são os mesmos definidos no Capítulo 3 para a proposta.

Uma informação importante é que, para cada cenário e execução, foi gerada uma semente para a criação de aleatórios, garantindo que a população inicial do AG em cada execução e no mesmo cenário seja sempre igual. Esta medida foi tomada para evitar que alguma execução fosse privilegiada com uma população inicial excepcionalmente boa.

4.3 Planejamento dos Experimentos

A comparação entre os trabalhos se dará aos pares, cenário a cenário, sempre entre os resultados da TBACO e os resultados de um dos três trabalhos comparados.

A comparação de apenas uma amostra de cada execução em cada cenário não é suficiente, pois este único valor pode não refletir a verdade.

A execução de um número maior de vezes, obtendo uma média e desvio padrão de cada população de resultados, permite uma comparação com melhor qualidade. Porém, se a diferença das médias e desvio padrão entre os trabalhos comparados não for grande o suficiente, não se pode garantir a constância dos resultados.

Tabela 4.3 Erros relativos obtidos pela primeira comparação entre trabalhos.

Cenário 15 x 15 / *Upperbound* = 1231 u.t.				
	Média	Desvio padrão	Erro Relativo	Dif. ER / Dif. Abs.
TBACO	1318,20	11,09	7,08	------
TGLOBAL	1332,26	15,19	8,23	1,15 / 14,06
TRN	1325,03	10,54	7,64	0,55 / 6,83
TCC	1325,46	9,85	7,67	0,59 / 7,26
Cenário 30 x 15 / *Upperbound* = 1764 u.t.				
	Média	Desvio padrão	Erro Relativo	Dif. ER / Dif, Abs.
TBACO	1956,86	12,53	10,93	-------
TGLOBAL	1985,28	21,56	12,54	1,61 / 28,43
TRN	1966,03	20,39	11,45	0,52 / 9,17
TCC	1963,97	21,32	11,34	0,40 / 7,11
Cenário 50 x 20 / *Upperbound* = 2868 u.t.				
	Média	Desvio padrão	Erro Relativo	Dif. ER / Dif. Abs.
TBACO	3237,43	26,17	12,88	------
TGLOBAL	3263,94	30,98	13,81	0,92 / 26,51
TRN	3251,91	26,43	13,39	0,51 / 14,49
TCC	3283,40	34,71	14,48	1,60 / 45,97

Na tabela acima, estão tabulados os dados resumidos da execução de 35 vezes de cada trabalho e cenário.

Upperbound é o menor valor possível para o *makespan* em um cenário, sendo o maior valor entre os somatórios de tempos de operação de cada tarefa e os somatórios de tempos de execução em cada máquina.

Erro relativo é o valor obtido pela equação:

$ER = ((makespan - upperbound)/upperbound) * 100.$

Vários trabalhos usam essa medida de desempenho nas suas validações.

A diferença de Erros Relativos é a diferença entre o erro relativo do trabalho em comparação e o erro relativo da TBACO.

A diferença absoluta é a diferença entre o *makespan* médio do trabalho em comparação e o *makespan* médio da TBACO.

Examinando esta tabela observe que não há como ter confiança na afirmação de que o método desenvolvido nesta tese é superior aos outros trabalhos, quando se pode perceber que todos os erros relativos são próximos e suas diferenças na maioria das vezes são inferiores a 1%. Também se percebe que as diferenças absolutas entre as médias, em sua maioria, estão abaixo dos desvios padrões médios das populações, permitindo pensar que em outro conjunto de resultados as situações se invertessem.

Portanto, para que se obtenha um número de execuções que garanta essa confiança, devem-se usar técnicas de estatística para calcular o número de amostras necessário para cada dupla de trabalhos comparados.

Entre as várias técnicas estatísticas disponíveis, escolheu-se o teste de hipóteses t-*student* por ser adequado ao problema (MONTGOMERY, 2005) e entre suas variações a que se aplica às médias de duas populações independentes e com variâncias desconhecidas.

O teste de hipóteses t-student fazendo inferência na diferença das médias de duas populações independentes com variâncias desconhecidas (ver Apêndice B) possui os seguintes requisitos:

- Distribuição normal;
- Populações independentes.

Considera-se a suposição de normalidade por conta do Teorema de Limite Central, que afirma que toda distribuição tende a normalidade na medida em que aumenta (MONTGOMERY, 2005) e que, usualmente, se costuma considerar população acima de 30 indivíduos como populações grandes (MONTGOMERY, 2009).

Consideram-se as populações como sendo independentes por serem geradas aleatoriamente (MONTGOMERY, 2005).

O teste necessita também da definição de dois parâmetros:

- o nível de significância, que é a margem assumida para o erro Tipo I; e
- o poder de teste, que é a margem assumida para o erro Tipo II.

Definiu-se o nível de confiança em 5%.

O poder de teste calculou-se em 95%, a ser garantido pelos tamanhos das populações amostrais a serem geradas na fase de cálculo de tamanho amostral.

Após gerarem-se as populações com o tamanho adequado e com tamanho mínimo de 35, para garantir a exigência de normalidade da distribuição (Teorema do Limite Central), aplica-se o teste de hipóteses t-student. A aplicação deste método está explicada no Apêndice B.

Com base nos resultados obtidos pela aplicação dos testes de hipóteses, pode-se concluir sobre a corroboração da hipótese e finaliza-se a validação.

Desta forma, os experimentos estão planejados nas fases apresentadas nas próximas três subseções.

4.3.1 **Geração de população inicial**

Nesta fase, se criam as populações iniciais dos quatro trabalhos nos três cenários. Cada população possui 35 indivíduos, número comumente usado nas análises do Tear.

Estas populações estão tabuladas no Apêndice C e uma tabela com as médias e desvios padrões é apresentada na execução desta fase.

4.3.2 **Cálculo do tamanho amostral e Complementação das populações**

Com as populações geradas na subseção anterior, faz-se o cálculo de tamanho amostral necessário para cada população para que se possa inferir com uma confiança razoável.

Como parâmetros para este cálculo são usados: nível de significância de 5%, poder de teste de 95% e tamanho do efeito. O tamanho de efeito é calculado especificamente para cada par. Os significados de tais parâmetros e as fórmulas que os equacionam estão no Apêndice B.

O cálculo do tamanho amostral é feito para cada par de trabalhos comparados. Parte-se das populações amostrais iniciais e calcula-se o tamanho amostral sugerido para estas populações. Se o valor obtido for menor ou igual ao limite de 200 elementos na amostra, considera-se o tamanho destas populações como o final, arredondados para cima. Caso seja maior considera-se o tamanho amostral como 200 e estima-se qual o poder de teste desta amostra.

Na fase de execução, uma tabela apresenta os principais resultados desta fase. O detalhamento dos cálculos assim como as populações finais usadas nos testes de hipótese estão no Apêndice C.

4.3.3 **Análise da hipótese**

Para cada par de trabalhos comparados, usou-se o teste de hipótese *t-student* fazendo inferência na diferença das médias para duas amostras independentes e com variâncias desconhecidas.

Este teste verificou se a hipótese de que o trabalho proposto nesta tese (TBACO) é melhor do que os trabalhos com os quais foi comparado (TGLOBAL, TRN e TCC). Para tal duas hipóteses nulas foram cunhadas:

- A média do *makespan* da população amostral de TBACO é menor ou igual a média do *makespan* da população amostral do outro trabalho na comparação;
- A média do *makespan* da população amostral de TBACO é igual à média do *makespan* da população amostral do outro trabalho na comparação.

Ao aceitarmos a primeira e refutarmos a segunda, corroboramos que a média do *makespan* da população amostral de TBACO é menor do que a média do *makespan* da população amostral do outro trabalho na comparação. Esta afirmação encontra-se comprovada no Apêndice B.

Uma tabela resumindo estas comparações é apresentada na seção que apresenta a execução desta fase.

4.4 Execução do Planejamento

Nesta seção, são apresentados os resultados da execução de cada fase do planejamento.

Quando for necessário, tabelas e figuras são apresentadas, porém de forma resumida. As tabelas analíticas estão expostas no Apêndice C.

A organização dessa seção segue a mesma estrutura da seção anterior, mas, agora, enfatizando os resultados obtidos na execução de cada fase.

Lembrando que as delimitações aos cenários propostos são:

- Cada máquina opera apenas uma tarefa por vez e cada tarefa só pode ser operada por uma máquina por vez;
- Os tempos de operação são determinados e previamente conhecidos;
- Os tempos de *setup* e transportes estão inclusos nos tempos de operação;
- Inicia-se o roteiro de uma tarefa transportando a matéria-prima do setor de carga para o *buffer* de entrada da primeira máquina em seu roteiro;
- Termina-se o roteiro de uma tarefa transportando-se a tarefa finalizada do *buffer* de saída da última máquina a operá-la para o setor de Descarga;
- Toda máquina possui *buffer* de entrada e *buffer* de saída de tamanho grande o suficiente para suportar qualquer demanda;
- Máquinas não param e nem dão defeito.

E que os parâmetros da colaboração entre Algoritmo Genético e VNS são:

- Tipo do cromossomo – baseado em operações;
- População de 30 indivíduos;
- Método de geração de população inicial - aleatório;
- Elitismo: 1 indivíduo;
- Seleção por Roleta: 9 indivíduos;
- Cruzamento – do tipo PPX com taxa de 66%;
- Mutação – usando operador 2-opt e taxa de 10%;
- Operadores de vizinhança - *Insert*, *Swap* e *2-opt* todos operando com posições aleatórias e operando 10 vezes cada.

4.4.1 **Geração de população inicial**

Nesta fase, criaram-se as populações iniciais dos quatro trabalhos nos três cenários.

Cada população possui 35 indivíduos, número comumente usado nas análises do Tear.

Os dados resumidos estão na Tabela 4.4 e os dados completos estão no Apêndice C, Seção Populações Iniciais.

Na Tabela 4.4 podem-se ver as médias dos *makespan* e o desvio padrão de cada população.

Tabela 4.4 Médias e desvios padrões das populações de 35 indivíduos para os quatro trabalhos nos três cenários da validação.

	Cenário	Média dos *makespans*	Desvio padrão
TBACO	15 x 15	**1318,20**	11,09
	30 x 15	**1956,86**	12,53
	50 x 20	**3237,43**	26,17
TGLOBAL	15 x 15	1332,26	15,19
	30 x 15	1985,28	21,56
	50 x 20	3263,94	30,98
TRN	15 x 15	1325,03	10,54
	30 x 15	1966,03	20,39
	50 x 20	3251,91	26,43
TCC	15 x 15	1325,46	9,85
	30 x 15	1963,97	21,32
	50 x 20	3283,40	34,71

Nesta tabela, já se percebe que o método proposto por esta tese possui melhores resultados médios em relação aos outros trabalhos, em todos os cenários usados.

4.4.2 **Cálculo do tamanho amostral e Complementação das populações**

Com as populações geradas na subseção anterior, inicia-se o cálculo de tamanho amostral necessário para cada população para que se possa inferir com uma confiança razoável. Esse cálculo é feito para cada par de trabalhos comparados.

Como parâmetros para este cálculo são usados: nível de significância de 5% (confiança de 95%) e poder de teste de 95%. Os significados de tais parâmetros e as fórmulas que os equacionam estão no Apêndice B.

Este cálculo ocorre com as populações iniciais de 35 elementos em cada amostra.

Se o valor encontrado for menor ou igual a 200, arredonda-se para cima, e considera-se como tamanho amostral para este par em comparação.

Se o valor encontrado for maior que 200, considera-se 200 como sendo o tamanho amostral e estima-se o poder de teste para este valor. Nos cálculos realizados nesta tese esta situação não ocorreu.

No Apêndice C todo o processo de geração dos tamanhos amostrais está explicado, com os dados gerados e tabelas minuciosas. As populações finais usadas nos testes de hipóteses também se encontram no Apêndice C.

Aqui se apresenta a Tabela 4.5 que resume os resultados finais dos cálculos amostrais realizados: os dados das populações (médias dos *makespans* e desvios padrões), os resultados para cada par em comparação (diferença entre médias, desvio padrão médio, efeito do tamanho e tamanhos amostrais calculados e usados).

A tabela é dividida em 3 seções, uma para cada cenário.

A primeira coluna indica com qual trabalho o TBACO está sendo comparado; a segunda e terceira colunas indicam a média de *makespan* e desvio padrão para o trabalho TBACO; a quarta e quinta colunas indicam a média do *makespan* e desvio padrão para o trabalho em comparação (aquele indicado na coluna um); a sexta coluna apresenta a diferença entre as médias das colunas dois e quatro; a coluna sete indica o desvio padrão médio dos desvios padrões apresentados nas colunas três e cinco; a coluna oito indica o efeito de tamanho calculado com a diferença e desvio médio das colunas seis e sete; a coluna nove apresenta o tamanho amostral calculado para este par de trabalhos e o tamanho amostral usado no teste de hipóteses.

Tabela 4.5 Resumo dos cálculos dos tamanhos amostrais para as comparações dos três trabalhos com o TBACO, nos três cenários dados.

Trabalho comparado	Trabalhos				Desvio padrão médio	Diferença das médias	Efeito de tamanho	Tamanho amostral calculado / usado
	TBACO		Comparado					
	Média	Desvio padrão	Média	Desvio padrão				
Cenário 15 x 15								
TGLOBAL	1318,20	11,09	1332,26	15,19	13,14	14,06	1,07	24 / 35
TRN	1316,43	12,31	1324,84	11,31	11,81	8,41	0,71	67 / 70
TCC	1316,48	12,17	1323,72	9,93	11,05	7,23	0,65	56 / 60
Cenário 30 x 15								
TGLOBAL	1956,86	12,53	1985,29	21,56	17,04	28,43	1,67	11 / 35
TRN	1955,36	14,50	1970,10	19,68	17,09	14,74	0,86	85 / 90
TCC	1954,52	15,50	1963,82	20,14	17,82	9,30	0,52	149 /150
Cenário 50 x 20								
TGLOBAL	3237,43	26,17	3263,94	30,98	28,57	26,51	0,93	32 / 35
TRN	3236,14	24,44	3248,77	27,96	25,85	12,62	0,49	87 / 90
TCC	3237,43	26,17	3283,40	34,71	30,44	45,97	1,51	13 / 35

Deve-se lembrar de que o cálculo de tamanho amostral visa indicar o tamanho mínimo que a amostra deve possuir para que se tenha um poder de teste desejado, nesta tese de 95%, e que considera-se como distribuição normal quando os tamanhos amostrais são superiores a 30.

4.4.3 **Análise das hipóteses**

Com os requisitos atendidos e as populações amostrais geradas, passa-se a realização dos testes de hipóteses.

Para cada par de trabalhos comparados, usou-se o teste de hipótese *t-student* fazendo inferência na diferença das médias para duas amostras independentes e com variâncias desconhecidas.

Este teste verificou se a hipótese de que o trabalho proposto nesta tese (TBACO) é melhor do que os trabalhos com os quais foi comparado (TGLOBAL, TRN e TCC).

A hipótese a ser testada é se a média dos *makespan* de TBACO é menor do que a média do *makespan* do trabalho com quem esta sendo comparado. Porém o suplemento usado para o cálculo do teste de hipóteses só permite as hipóteses nulas na forma: igual, maior igual e menor igual.

Portanto duas hipóteses nulas foram cunhadas:

- A média do *makespan* da população amostral de TBACO é menor ou igual a média do *makespan* da população amostral do outro trabalho na comparação;
- A média do *makespan* da população amostral de TBACO é igual à média do *makespan* da população amostral do outro trabalho na comparação.

A hipótese nula é rejeitada sempre que o p-valor for menor que 0,05.

Ao se aceitar a primeira hipótese nula e se refutar a segunda hipótese nula, implica em se aceitar que a média do *makespan* da população amostral de TBACO é menor do que a média do *makespan* da população amostral do outro trabalho na comparação. Esta afirmação encontra-se comprovada no Apêndice B.

A Tabela 4.6 resume os resultados finais dos testes de hipóteses realizados: os dados das populações (médias dos *makespans* e desvios padrões), os tamanhos amostrais usado e os p-valores para as duas hipóteses nulas testadas.

A tabela é dividida em 3 seções, uma para cada cenário, e as colunas apresentam as seguintes informações:

- A primeira coluna indica com qual trabalho o TBACO está sendo comparado;
- A segunda e a terceira colunas indicam a média de *makespan* e desvio padrão para o trabalho TBACO;
- A quarta e a quinta colunas indicam a média do *makespan* e desvio padrão para o trabalho em comparação (aquele indicado na coluna um);
- A sexta coluna apresenta o tamanho das amostras do par sendo comparado;
- A sétima coluna indica o p-valor calculado para a hipótese nula que diz que a média dos *makespan* do TBACO é menor ou igual à média dos *makespan* do trabalho sendo comparado;

Tabela 4.6 Resumo dos testes de hipóteses para as comparações dos três trabalhos com o TBACO, nos três cenários dados.

Trabalho comparado	Trabalhos				Tamanho amostral usado	p-valor para TBACO <=	p-valor para TBACO =
	TBACO		Comparado				
	Média	Desvio padrão	Média	Desvio padrão			
Cenário 15 x 15							
TGLOBAL	1318,20	11,09	1332,26	15,19	35	0,999980018	4,00E-05
TRN	1316,43	12,31	1324,84	11,31	70	0,999977072	4,59E-05
TCC	1316,48	12,17	1323,72	9,93	60	0,999735584	0,000528832
Cenário 30 x 15							
TGLOBAL	1956,86	12,53	1985,29	21,56	35	0,999999995	1,02E-08
TRN	1955,36	14,50	1970,10	19,68	90	0,999999976	4,90E-08
TCC	1954,52	15,50	1963,82	20,14	150	0,999994605	1,08E-05
Cenário 50 x 20							
TGLOBAL	3237,43	26,17	3263,94	30,98	35	0,999873407	0,000253185
TRN	3236,14	24,44	3248,77	27,96	90	0,999354911	0,001290177
TCC	3237,43	26,17	3283,40	34,71	35	0,999999981	3,83E-08

- A oitava coluna indica o p-valor calculado para a hipótese nula que diz que a média dos *makespan* do TBACO é igual à média dos *makespan* do trabalho sendo comparado.

Analisando-se a Tabela 4.6 verifica-se que:

- Quando a hipótese nula é que o *makespan* médio de TBACO é igual ao *makespan* médio de qualquer um dos trabalhos em comparação, o p-valor é bem menor que 0,05, caracterizando a não aceitação desta hipótese nula; e
- Quando a hipótese nula é que o *makespan* médio de TBACO é menor ou igual ao *makespan* médio de qualquer um dos trabalhos em comparação, o p-valor aproxima-se de um, caracterizando a aceitação desta hipótese nula.

Para todas as comparações feitas nota-se que o *makespan* médio de TBACO é menor que o *makespan* médio de qualquer um dos trabalhos em comparação.

Portanto, estatisticamente, com confiança de 95%, usando populações adequadas (pois possuem distribuição normal e são independentes), o método desta tese (TBACO) é corroborado como sendo melhor que os métodos TGLOBAL, TRN e CC nos três cenários e contexto dados.

4.5 Gráficos e tabelas adicionais

No Apêndice D são apresentados 12 gráficos mostrando a convergência do *makespan* para cada método comparado, em cada cenário. Cada gráfico apresenta a convergência da média do pior *makespan*, a média do *makespan* médio e a média do melhor *makespan* de cada geração.

No Apêndice D são apresentados 3 gráficos mostrando a convergência dos 4 métodos, um por cenário. Cada gráfico apresenta as convergências ao longo do tempo de cinco minutos. A curva indica a média do melhor *makespan* no tempo indicado.

Estes gráficos apresentados no Apêndice D servem de apoio para os trabalhos futuros e são comentados oportunamente no Capítulo 5.

No Apêndice D são apresentados também tabelas mostrando o desempenho dos operadores de vizinhança, tanto a nível de melhoria da solução (o operador que obteve a melhor solução na busca local efetuada) quanto a nível de cada operação de vizinhança. Os resultados são analisados e comentados no Capítulo 5, trabalhos futuros.

4.6 Considerações finais

Neste capítulo, foi apresentado o contexto da validação (definindo-se os cenários, o problema a ser tratado e os trabalhos a serem comparados), o planejamento dos experimentos

(compreendendo desde a geração inicial de populações até a análise de hipótese) e a execução dos experimentos planejados (com a análise estatística dos dados obtidos levando-se em consideração a diferença de suas médias).

Ao final, pode-se afirmar que a proposta desta tese está corroborada, pois não se conseguiu falsear a hipótese de que esta seja melhor do que os trabalhos usados na comparação, no contexto definido.

Capítulo 5

CONCLUSÃO

5.1 Síntese do contexto da proposta e dos objetivos

Na indústria manufatureira, ocorrem imprevistos durante a execução de uma programação das operações, que é gerada por programação da produção em tempo de planejamento, criando a urgência em se obter nova programação dentro de um limite de tempo considerado curto (PINEDO, 2005).

Visando conseguir resultados melhores aos obtidos pelos métodos atualmente em uso, esta tese propôs uma colaboração entre AG e VNS, usando como estrutura de vizinhança, em tempo de busca local, uma nova estrutura baseada em árvore de cobertura voltada a minimizar o *makespan* em problemas de Programação Reativa da Produção.

5.2 Contribuições e delimitações

Esta proposta de tese de doutorado foi desenvolvida considerando as delimitações listadas na Seção 1.5 e validada usando cenários de *job shop* de três diferentes dimensões, buscando a minimização de *makespan* dentro de um tempo de processamento de cinco minutos.

Foi feita uma comparação do método proposto nesta tese, nomeado TBACO, com três trabalhos (AG sem busca local, AG com VNS usando RN e AG com VNS usando CC) e os resultados corroboraram a hipótese de que uma estrutura de vizinhança mais correlata ao *makespan* aplicada em uma colaboração de AG com VNS obtém resultados melhores aos obtidos por outros métodos usados atualmente. Deve-se lembrar que o tempo de processamento da busca não pôde ultrapassar um tempo limite (no caso dos testes, foram cinco minutos) e este limite influencia diretamente a qualidade obtida nos *makespans* durante os experimentos.

As amostras foram produzidas em número suficiente para garantir atendimento a Teoria do Limite Central (> 30 elementos permite considerar como distribuição normal) e ao poder de teste desejado (95%). Estas amostras são comparadas usando teste de hipótese de t-

student por inferência nas médias de *makespan* com 95% de confiança. A hipótese testada foi: a média dos *makespans* obtida por TBACO é melhor (menor) do que a média dos *makespans* obtida por cada um dos outros 3 trabalhos em comparação (TGLOBAL, TRN e TCC). Por questões práticas, foram formuladas duas hipóteses nulas, a primeira TBACO <= (TGLOBAL,TRN e TCC) e a segunda TBACO = (TGLOBAL, TRN e TCC). Ressalta-se que *t-student* forneceu um p-valor indicando a relevância do resultado e, usualmente quando este valor é inferior a 0,05 significa que se pode rejeitar a hipótese nula com 95% de confiança.

Os valores de p-valor destes testes estão apresentados na Tabela 4.6.

Para todos os cenários percebeu-se que TBACO obteve p-valores sempre superiores a 0,05 quando a proposta nula testada afirmava que TBACO possui resultados menores ou iguais aos dos outros trabalhos, sendo aceita com 95% de chances de a proposta produzir resultados melhores do que os trabalhos comparados.

Percebeu-se também que para todos os cenários TBACO obteve p-valores sempre inferiores a 0,05 quando a proposta nula testada afirmava que a proposta e os concorrentes possuiam resultados equivalentes, sendo rejeitada com 95% de chances de a proposta produzir resultados equivalentes aos trabalhos comparados.

Feita esta análise, pôde-se afirmar que a proposta conseguiu, com 95% de confiança e 95% de poder de teste, ser melhor aos trabalhos comparados, dentro dos contextos dados e das delimitações impostas, considerando como tempo limite de processamento cinco minutos.

Como estes contextos, delimitações e limite de tempo aproximam este trabalho da realidade, pôde-se afirmar que a principal contribuição desta tese de doutorado está na melhoria do *makespan* encontrado para problemas de PRP por meio da criação de um novo método de busca competitivo com os métodos atuais, na procura por menor *makespan* em problemas de Programação Reativa da Produção.

Como *trade-off* pode-se dizer que o BACO sacrificou a simplicidade da RN e a alta correlação do CC em troca de um desempenho superior a estas duas.

Uma outra constatação é que pela análise dos gráficos de convergência geração a geração (Apêndice D, seção D.1), de cada método e em cada cenário, pode-se depreender que o AG converge em todos de forma semelhante, porém o método BACO tira melhor proveito destas gerações.

5.3 Trabalhos futuros

Durante a elaboração desta tese, tanto na fase de definição da proposta quanto na fase de validação, várias observações foram feitas e as que são mais pertinentes como propostas de trabalhos futuros são:

- Embora o problema foco seja PRP, não há motivo para não se estudar a aplicação de suas possibilidades para problemas de PP, transferindo a contribuição desta tese para outra área afim. O motivo de sua possível boa aplicação se deve ao fato do método criado nesta tese ter sido testado no que se pode chamar de PP com pouco tempo de cálculo, visto que a validação se deu desse modo;
- Mais um estudo que poderia derivar deste é a determinação de estruturas correlatas a outros objetivos de programação, tais como tempo devido, lucro estimado, multa por atraso, entre outros. Note-se que a estrutura proposta é altamente correlata ao *makespan,* porque se correlacionam os tempos de término das operações da programação das operações aos tempos de fim das transições disparadas (operações) ao longo do caminho na árvore de cobertura;
- Embora a área de aplicação seja a indústria manufatureira e, para tal aplicação, o limite de tempo estipulado está adequado, o método proposto poderia ser estudado para ser empregado em outras áreas tal como atribuição de tarefas em sistemas operacionais ou equivalentes, com o objetivo de minimizar makespan;
- Da observação dos gráficos do Apêndice D - Seção 2, percebe-se que a partir de certo tempo, na média em torno dos 80% do tempo limitado ao processamento, esta proposta tem o melhor resultado e não perde mais para os outros trabalhos. Baseado nesta constatação, seria interessante estudar a aplicação de variações nos algoritmos de busca, tanto global como local, a partir deste ponto, visando explorar o espaço de busca diferentemente do que foi definido no primeiro intervalo de tempo;
- Uma observação feita nas tabelas de desempenho do Apêndice D – Seção 3 é a de que o operador *2-opt* opera a maioria das melhoras nas buscas locais, em torno de 44%, permitindo pensar em um estudo em que só se trabalhe com este operador, talvez aumentando o número de operações específicas com ele e diminuindo o número de operações no geral, permitindo reduzir o tempo da busca local e aumentar o número de gerações do AG no tempo limite;

Por fim, além das considerações anteriores provenientes de observações feitas durante este estudo, outras pesquisas podem ser sugeridas. Algumas baseadas no relaxamento de

várias das limitações impostas no contexto desta tese, tais como uso de AGV, tempo de espera entre tarefas, preempção e tamanho de *buffers* de entrada e saída, ou ainda, pela aplicação em outros problemas além do *job shop*, tais como *open shop* ou *flow shop* e suas variantes.

Embora a proposta BACO tenha contemplado uma estrutura virtual em seu bojo, não foi verificado o quanto esta estrutura influenciou nos seus resultados, podendo ser feita uma análise futura re-executando-se os testes com e sem esta estrutura.

REFERÊNCIAS

AARTS, E. H. L.; LENSTRA, J. K. Local Search in Combinatorial Optimization. 2. ed. Princeton: University Press, 2004.

ANDERSON, D. R. Estatística Aplicada à Economia. 1. ed. São Paulo: Pioneira Thomson Learning, 2002.

ARENALES, M.; ARMENTATO, V.; MORABITO, R.; YANASSE, H. Pesquisa Operacional. 4. ed. Rio de Janeiro: Campus-Elsevier, 2007.

BEASLEY, J. E. OR-library: Distributing test problems by electronic mail. Journal of Operacional Research Society, v. 41, n. 11, p. 1069-1072., 1990.

BRANDIMARTE, P. Routing and scheduling in a flexible job shop by taboo search. Annals of operations research, v. 41, n. 3, p. 157-183, 1993

CHENG, R.; GEN, M; TSUJIMURA, Y. A tutorial survey of job shop scheduling problems using genetic algorithms I - representation. Computers & Industrial Engineering., v. 30, n. 4, p. 983 – 997, 1996.

CHOI, C.; LEE, J. Chaotic local search algorithm. Artificial Life Robotics, v. 2, p. 41-47, 1998.

DAUZÉRE-PÉRÉS, S.; PAULLI, J. An integrated approach for modeling and solving the general multiprocessor job shop scheduling problem using tabu search. Annals of Operations Research, v. 70, n. 3, p. 281-306, 1997.

DEMSAR, J. Statistical comparisons of classifiers over multiple data sets. Journal of Machine Learning Research, v. 7, p. 1-30, 2006.

DUAN, J.; PAN, Q.; LI, J.; GAO, K.; BAO, Z. Scheduling the Blocking Flow Shop Problem Using a Harmony Search Algorithm. In: Sixth International Conference on Natural Computation (ICNC), 2010, China, Proceedings of Sixth International Conference on Natural Computation (ICNC), v. 8, p. 4258-4262, 2010.

ESWARAMURTHY, V. P.; TAMILARASI, A. Hybridizing tabu search with ant colony optimization for solving job shop scheduling problems. International Journal of Advanced Manufacturing Technology, v. 40, p. 1004-1015, 2009.

FALKENAUER, E.; BOUFFOUIX, S. A genetic algorithm for job shop. Proceedings of the IEEE International Conference on Robotics and Automation, 1991.

GAO, L.; ZHANG, G.; ZHANG, L.; LI, X. An efficient memetic algorithm for solving the job shop scheduling problem. Computers and Industrial Engineering. v. 60, p. 699-705, 2011.

GILAK, E.; RASHIDI, H. A new hybrid Electromagnetism Algorithm for Job Shop Scheduling. In: Third UKSim European Symposium on Computer Modeling and Simulation, 2009, Grécia, Proceedings of Third UKSim European Symposium on Computer Modeling and Simulation, v. 1, p. 327-332, 2009.

GLOVER, F. Genetic algorithms and Scatter search: unsuspected potentials. Statistics and Computing, v. 4, p. 131-140, 1994.

HAN, Y.; DUAN, J.; ZHANG, M. Apply the discrete artificial bee colony algorithm to the blocking flow shop problem with makespan criterion. In: Chinese Control and Decision Conference (CCDC), 2011, China, Proceedings of Chinese Control and Decision Conference (CCDC), v. 1, p. 2131-2135, 2011.

HANSEN, P.; MLADENOVICT, N.; PÉREZ, J. A. M. Variable Neighbourhood Search: Methods and applications. Annals of Operations Research, v. 175, p. 367-407, 2010.

HELLER, J. Some numerical experiments for an M x J flowshop and its decision-theoretical aspects. Operational Research., v. 8, p.178-184, 1960.

HMIDA, A. B.; HAOUARI, M.; HUGUET, M.; LOPEZ, P. Discrepancy search for the flexible job shop scheduling problem. Computers & Operations Research, v. 37, p. 2192-2201, 2010.

HOLLAND, J. H. Adaptation in Natural and Artificial Systems. 2. ed. Ann Arbor: University of Michigan Press, 1975.

HURINK, E.; JURISCH, B.; THOLE, M. Tabu search for job shop scheduling problem with multi-purpose machine. Operations Research Spektrum, v. 15, p. 205-215, 1994.

JAIN, A.; JAIN, P. K.; SINGH, I. P. Performance modeling of FMS with flexible process plans – a Petri Net approach. International Journal of Simulation Modelling, v. 5, p. 101-113, 2006.

JARBONI, B.; EDDALY, M.; SIARRY, P. A hybrid genetic algorithm for solving no-wait flowshop scheduling problems. International Journal of Advanced Manufacturing Technology, v. 54, p. 1129-1143, 2011.

KACEM, L.; HAMMADI, S.; BORNE, P. Approach by localization and multiobjective evolucionary optimization for flexible job shop scheduling problems. IEEE Transactions on Systems, Man and Cybernetics, part c, v. .32, n. 1, p. 1-13, 2002a.

KACEM, L.; HAMMADI, S.; BORNE, P. Pareto-optimality approach for flexible job shop scheduling problems: hybridization of evolucionary algorithms and fuzzy logic. Mathematics and Computers in Simulation, v. 60, p. 245-276, 2002b.

KENNEDY, J.; EBERHART, R.C. Particle Swarm Optimization. In Proceedings of 1995 IEEE International Conference on Neural Networks, Piscataway, NJ: IEEE Service Center, v. 4, p. 1942-1948, 1995.

K-H, H. Genetic Quantum Algorithm and its application combinatorial optimization problems. In: IEEE Proceedings of the 2000 Congress on Evolutionary Computation, San Diego, USA IEEEE Press, 2000.

KIM, W. Y.; SUZUKI, T.; NARIKIYO, T. FMS based on timed Petri net model and reactive graph search. Applied Mathematical Modeling, v. 31, p. 955-970, 2007.

LAKATOS, E. M.; MARCONI, M. A. Metodologia Científica. 5. ed. São Paulo: Atlas, 2008.

LI, J.; PAM, Q.; SUGANTHAN, P. N.; CHUA, T. J. A hybrid tabu search algorithm with an efficient neighborhood structure for the flexible job shop scheduling problem. International Journal of Advanced Manufacturing Technology, v. 52, p. 683-697, 2011.

MATTFELD, D. C.; BIERWIRTH, C. An Efficient Genetic Algorithm for Job Shop Scheduling with Tardiness Objectives, European Journal of Operational Research, v. 155, p. 616 – 630, 2004.

MCCLAVE, J. T. Estatística para Administração e Economia. 1. ed. São Paulo: Pearson Prentice Hall, 2009.

MEJIA, G.; ODREY, N. G. An approach using Petri nets and improved heuristic search for manufacturing system scheduling. Journal of Manufacturing Systems, v.24, p. 79-92, 2005.

MIRSANEI, H. S.; ZANDIEH, M.; MOAYED, M. J.; KHABBAZI, M. R. A simulated annealing algorithm approach to hybrid flow shop scheduling with sequence-dependence setup times. Journal of Intelligent Manufacturing, v. 22, p. 965-978, 2011.

MLADENOVICT, N; HANSEN, P. Variable Neighborhood Search. Computers and Operations Research, v. 24, p. 1097-1100, 1997.

MONTGOMERY, D. C. Design and Analysis of Experiments. 6. ed. New York: John Wiley, 2005.

MONTGOMERY, D. C.; RUNGER, G. C. Estatística aplicada e probabilidade para engenheiros. 4. ed. Rio de Janeiro: LTC, 2009.

MORANDIN JR, O.; CARIDA, V. F.; KATO, E. R. R.; TUMA, C. C. M. Adaptive Genetic Fuzzy, Predictive and Multiobjective Approach for AGVs Dispatching. In: IECON 2011 - 37th Annual Conference of the IEEE Industrial Electronics Society, 2011, Australia, Proceedings of the IECON 2011 - 37th Annual Conference of the IEEE Industrial Electronics Society, v. 1 , p. 2317-2322, 2011.

MORANDIN JR, O.; KATO, E. R. R.; ARAUJO, R. G.; VIEIRA, L. S. A Modeling Strategy for Control and Interlocking of an AMS using Virtual Petri Nets. In: SMC2007 - IEEE International Conference on Systems, Man, and Cybernetics, 2007, Canadá, Proceedings of the SMC2007 - IEEE International Conference on Systems, Man, and Cybernetics, v. 1, p. 3493-3498, 2007b.

MORANDIN JR, O.; KATO, E. R. R.; MAGGIO, E. G. R.; SANCHES, D. S.; DERIZ, A. C. A Heuristic based on Petri Nets modeling for FMS Scheduling problem of makespan minimization. In: IECON 2007 - 33rd Annual Conference of IEEE Industrial Electronics, 2007, Taiwan, Proceedings of the IECON 2007 - 33rd Annual Conference of IEEE Industrial Electronics, v. 1, p. 2683-2688, 2007a.

MORANDIN JR, O.; KATO, E. R. R.; TUMA, C. C. M. A strategy of production scheduling with the fitness function of genetic algorithm using Timed Petri net and considering AGV and the input buffer. In: IECON 2010 - 36th Annual Conference of the IEEE Industrial Electronics Society, 2010, EUA, Proceedings of the IECON 2010 - 36th Annual Conference of the IEEE Industrial Electronics Society, v. 1. p. 1311-1316, 2010.

MORETTIN, P. A.; BUSSAB, W. O. Estatística básica. 5. ed. São Paulo: Saraiva, 2002.

MORO, A. R.; YU, H.; KELLEHER, G. Hybrid heuristic search for the scheduling of flexible manufacturing systems using Petri nets. IEEE Transactions on Robotics and Automation, v. 18, p. 240-245, 2002.

MURATA, T. Petri Nets: Properties, Analysis and Applications. Proceedings of the IEEE, v. 77, n. 4, p. 541-560, 1989.

NAWAZ, M.; ENSCORE JR, E. E.; HAM, I. A heuristic algorithm for the m-machine, n-job flow-shop sequencing problem, Omega, Elsevier, v. 11, p. 91-95, 1983.

PAN, Q.; WANG, L.; GAO, L.; LI, J. An effective shuffled frog-leaping algorithm for lot-streaming flow shop scheduling problem. International Journal of Advanced Manufacturing Technology, v. 52, p. 699-713, 2011.

PAN, Q.; WANG, L.; GAO, L.; LI, W. D. An effective hybrid discrete differential evolution algorithm for the flow shop scheduling with intermediate buffers. Information Sciences, v. 181, p. 668-685, 2011.

PINEDO, M. L. Planning and Scheduling in Manufacturing and Services. New York: Springer, 2005.

POPPER, K. R. A lógica da pesquisa científica. 20. ed. São Paulo: Cultrix, 2007.

PORTAL ACTION Action v2.8. Disponível em: <http://www.portalaction.com.br/content/download-action->. Acesso em 20/01/2015.

POTVIN, J. Y. Genetic Algorithms for the Traveling Salesman Problem. Annals of Operations Research v. 63, p. 339–370, 1996.

REEVES, C. R. A genetic algorithm for flowshop sequencing. Computers & Operations Research., v. 22, p.5-13, 1995.

RIBAS, I.; COMPANYS, R.; TORT-MARTORELL, X. An iterated greedy algorithm for the flowshop scheduling problem with blocking. Omega, v. 39, p. 293-301, 2011.

ROSHANAEI, V.; NADERI, B.; JOLAI, F.; KHALILI, M. A variable neighborhood search for job shop scheduling with set-up times to minimize makespan. Future Generation Computer Systems, v. 25, p. 654-661, 2009.

SELS, V.; CRAEYMEERSCH, K.;VANHOUCKE, M. A hybrid single and dual population search procedure for the job shop scheduling problem. European Journal of Operational Research, v. 215, p. 512-523, 2011.

SONG, L.; XU, X. Flexible Job Shop Scheduling Problem Solving Based on Genetic Algorithm with Chaotic Local Search. In: Sixth International Conference on Natural Computation (ICNC 2010), 2010, Guangzhou, China. Proceedings of Sixth International Conference on Natural Computation (ICNC 2010), v. 5, p. 2356-2360, 2010.

STORN, R.; PRICE, K. Differential Evolution: a simple and efficient adaptative scheme for global optimization over continuous spaces. Journal of Global Optimization, v. 11, p. 341-359, 1997.

SUN, L.; LIU, M.; ZHANG, C. Y.; GAO, L.; LIAN, K. L. New High Performing Hybrid Particle Swarm Optimization for Permutation Flow Shop Scheduling Problem with Minimization of Makespan. In: International Conference on Industrial Engineering Management (IEEM), 2010, China, Proceedings of International Conference on Industrial Engineering Management (IEEM), v. 1, p. 1706-1710, 2010.

TAHA, H. A. Pesquisa Operacional. 8. ed. São Paulo: Pearson Prentice-Hall, 2008.

TAILLARD, E. Problems instances. Disponível em: <http://mistic.heig-vd.ch/taillard/problemes.dir/- >. Acesso em: 20/01/2015.

TAILLARD, E. Benchmarks for basic scheduling problems. European Journal of Operational Research, v. 64, p. 278-285, 1993.

TANG, J.; ZHANG, G.; LIN, B.; ZHANG, B. A Hybrid Algorithm for Flexible Job-Shop Scheduling Problem. Procedia Engineering, v. 15, p. 3678-3683, 2011.

TAVARES, L. V., CORREIA, F. N. Optimização linear e não linear: conceitos, métodos e algoritmos. 2. ed. Lisboa: Fundação Calouste Gulbenkian, 1999.

TRAN, T. H.; NG, K. M. A water-flow algoritm for flexible flow shop scheduling with intermediate buffers. Journal of Scheduling, v. 14, p. 483-500, 2011.

TRIOLA, M. F. Introdução à Estatística. 10. ed. Rio de Janeiro: LTC, 2008.

VELA, R. C.; VARELA, R.; GONZÁLEZ, M. A. Local search and genetic algorithm for the job shop scheduling problem with sequence dependent setup times. Journal of Heuristics v. 16, p. 136-165, 2010

WANG, B.; ZHANG, G. Hybrid VNS and Memetic Algorithm for Sovling The Job Shop Scheduling Problem. In: 18th International Conference on Industrial Engineering and engineering Management (IE&EM), 2011, China, Proceedings of 18th International Conference on Industrial Engineering and engineering Management (IE&EM), v. 2, p. 924-927, 2011.

WANG, L.; ZHOU, G.; XU, Y.; WANG, S.; LIU, M. An effective artificial bee colony algorithm for the flexible job-shop scheduling problem. International Journal of Advanced Manufacturing Technology, v. 60, p. 303-315, 2012.

YAZDANI, M.; AMIRI, M.; ZANDIEH, M. Flexible job-shop with parallel variable neighborhood search algorithm. Expert Systems with applications, v. 37, p. 678-687, 2010.

ZHANG, J.; ZHANG, C.; LIANG, S. The circular discrete particle swarm optimization algorithm for flow shop scheduling problem. Expert Systems with applications, v. 37, p. 5827-5834, 2010.

ZHANG, X.; KOSHIMURA, M.; FUJITA, H.; HASEGAWA, R. Combining PSO and Local Search to Solve Scheduling Problems. In: 13th Annual Conference Companion on Genetic and evolutionary Computation GECCO '11, 2011, USA, Proceedings of 13th Annual Conference Companion on Genetic and evolutionary Computation GECCO '11, v. 1, p. 347-354, 2011.

ZHANG, Z.; ZHANG, J.; LI, S. A Modified Ant Colony Algorithm for the Job Shop Scheduling Problem to Minimize Makespan. In: International Conference on Mechanic Automation and Control Engineering (MACE), 2010, China, Proceedings of International Conference on Mechanic Automation and Control Engineering (MACE), v. 1, p. 3627-3630, 2010.

ZHENG, T.; YAMASHIRO, M. Solving flow shop scheduling problems by quantum differential evolutionary algorithm. International Journal of Advanced Manufacturing Technology, v. 49, p. 643-662, 2010.

ZOBOLAS, G. I.; TARANTILIS, C. D.; IOANNOU, G. Minimizing makespan in permutation flow shop scheduling problems using a hybrid metaheuristic algorithm. Computers & Operations Research, v. 36, p. 1249-1267, 2009.

.

Apêndice A

CENÁRIOS

Neste apêndice, estão as tabelas apresentando os cenários usados na validação desta tese de doutorado.

As tabelas A.1 e A.2 apresentam os dados referentes ao cenário de *job shop* de 15 tarefas por 15 máquinas, importado de ta01 da base de cenários de (TAILLARD, 2015).

As tabelas A.3 e A.4 apresentam os dados referentes ao cenário de *job shop* de 30 tarefas por 15 máquinas, importado de ta31 da base de cenários de (TAILLARD, 2015).

As tabelas A.5 e A.6 apresentam os dados referentes ao cenário de *job shop* de 50 tarefas por 20 máquinas, importado de ta61 da base de cenários de (TAILLARD, 2015).

Cada linha representa o roteiro de uma tarefa e cada coluna representa a máquina responsável por executar a operação nessa fase (tabelas A.1, A.3 e A.5) ou o tempo determinístico que leva essa operação nessa fase (tabelas A.2, A.4 e A.6).

Tabela A.1 Roteiros de máquinas do cenário de *job shop* 15 tarefas x 15 máquinas(ta01).

	Máquina responsável pela etapa da tarefa														
Tarefa 1	7	13	5	8	4	3	11	12	9	15	10	14	6	1	2
Tarefa 2	5	6	8	15	14	9	12	10	7	11	1	4	13	2	3
Tarefa 3	2	9	10	13	7	12	14	6	1	3	8	11	5	4	15
Tarefa 4	6	3	10	7	11	1	14	5	8	15	12	9	13	2	4
Tarefa 5	8	9	7	11	5	10	3	15	13	6	2	14	12	1	4
Tarefa 6	6	4	13	14	12	5	15	8	3	2	11	1	10	7	9
Tarefa 7	13	4	8	9	15	7	2	12	5	6	3	11	1	14	10
Tarefa 8	12	6	1	8	13	14	15	2	3	9	5	4	10	7	11
Tarefa 9	11	12	7	15	1	2	3	6	13	5	9	8	10	14	4
Tarefa 10	7	12	10	3	9	1	14	4	11	8	2	13	15	5	6
Tarefa 11	5	8	14	1	6	13	7	9	15	11	4	2	12	10	3
Tarefa 12	3	15	1	13	7	11	8	6	9	10	14	2	4	12	5
Tarefa 13	6	9	11	3	4	7	10	1	14	5	2	12	13	8	15
Tarefa 14	9	15	5	14	6	7	10	2	13	8	12	11	4	3	1
Tarefa 15	11	9	13	7	5	2	14	15	12	1	8	4	3	10	6

Tabela A.1 Tempos de operações do cenário de *job shop* 15 tarefas x 15 máquinas(ta01).

	Tempo de duração de cada etapa da tarefa														
Tarefa 1	94	66	10	53	26	15	65	82	10	27	93	92	96	70	83
Tarefa 2	74	31	88	51	57	78	8	7	91	79	18	51	18	99	33
Tarefa 3	4	82	40	86	50	54	21	6	54	68	82	20	39	35	68
Tarefa 4	73	23	30	30	53	94	58	93	32	91	30	56	27	92	9
Tarefa 5	78	23	21	60	36	29	95	99	79	76	93	42	52	42	96
Tarefa 6	29	61	88	70	16	31	65	83	78	26	50	87	62	14	30
Tarefa 7	18	75	20	4	91	68	19	54	85	73	43	24	37	87	66
Tarefa 8	32	52	9	49	61	35	99	62	6	62	7	80	3	57	7
Tarefa 9	85	30	96	91	13	87	82	83	78	56	85	8	66	88	15
Tarefa 10	5	59	30	60	41	17	66	89	78	88	69	45	82	6	13
Tarefa 11	90	27	1	8	91	80	89	49	32	28	90	93	6	35	73
Tarefa 12	47	43	75	8	51	3	84	34	28	60	69	45	67	58	87
Tarefa 13	65	62	97	20	31	33	33	77	50	80	48	90	75	96	44
Tarefa 14	28	21	51	75	17	89	59	56	63	18	17	30	16	7	35
Tarefa 15	57	16	42	34	37	26	68	73	5	8	12	87	83	20	97

Tabela A.2 Roteiros de máquinas do cenário de *job shop* 30 tarefas x 15 máquinas(ta31).

	Máquina responsável pela etapa da tarefa														
Tarefa 1	4	11	15	2	6	9	5	12	14	13	3	10	8	1	7
Tarefa 2	7	5	3	4	15	6	8	14	10	1	12	13	9	11	2
Tarefa 3	5	3	11	6	13	14	4	12	10	8	9	7	15	2	1
Tarefa 4	1	14	5	2	15	10	13	4	9	6	3	7	11	8	12
Tarefa 5	7	2	11	5	9	6	3	10	8	1	14	12	4	15	13
Tarefa 6	6	7	13	4	1	5	3	9	2	14	12	10	15	8	11
Tarefa 7	12	14	2	7	5	13	15	1	11	9	6	8	3	10	4
Tarefa 8	6	3	2	5	10	14	9	11	12	4	8	1	7	13	15
Tarefa 9	8	14	13	6	11	10	1	12	9	5	3	2	15	7	4
Tarefa 10	7	8	15	2	10	12	4	6	11	3	9	1	14	13	5
Tarefa 11	11	1	4	5	6	12	8	13	14	15	9	2	7	3	10
Tarefa 12	6	1	10	12	13	7	11	15	9	5	8	4	14	2	3
Tarefa 13	9	1	8	5	4	2	14	6	7	13	12	10	3	15	11
Tarefa 14	5	10	14	13	7	3	8	2	12	11	9	15	1	6	4
Tarefa 15	8	15	12	3	11	13	2	4	14	10	5	9	6	1	7
Tarefa 16	14	13	5	12	2	1	11	7	6	10	3	8	4	15	9
Tarefa 17	1	8	13	15	4	3	9	12	14	10	5	2	6	7	11
Tarefa 18	6	7	8	5	13	10	12	4	11	9	2	1	3	15	14
Tarefa 19	13	5	4	14	12	7	6	1	11	2	3	10	8	9	15
Tarefa 20	1	8	4	12	11	2	9	13	6	7	3	15	10	14	5
Tarefa 21	15	6	8	2	11	7	10	4	13	1	12	14	5	3	9
Tarefa 22	13	12	7	9	14	11	2	8	15	10	5	4	3	1	6
Tarefa 23	9	10	12	4	5	14	11	3	1	13	6	7	8	2	15
Tarefa 24	13	9	7	10	12	6	3	8	15	1	5	2	4	14	11
Tarefa 25	15	12	9	5	11	6	4	3	7	10	13	14	1	2	8
Tarefa 26	8	15	5	1	13	11	9	6	4	2	7	10	3	12	14
Tarefa 27	12	13	5	15	14	2	6	9	1	8	11	10	3	7	4
Tarefa 28	15	9	8	2	1	7	10	13	6	11	5	14	12	3	4
Tarefa 29	9	8	11	15	6	13	10	3	12	2	4	1	14	7	5
Tarefa 30	3	13	14	4	1	15	7	6	11	12	9	10	2	8	5

Tabela A.3 Tempos de operações do cenário de *job shop* 30 tarefas x 15 máquinas(ta31).

	Tempo de duração de cada etapa da tarefa														
Tarefa 1	99	43	6	99	23	98	84	24	30	53	34	95	50	48	38
Tarefa 2	19	24	65	16	94	9	60	32	59	85	9	36	22	25	5
Tarefa 3	54	62	93	78	59	71	49	88	40	13	17	88	47	30	56
Tarefa 4	60	16	79	84	84	42	59	14	74	60	98	17	42	31	19
Tarefa 5	49	52	46	50	1	14	2	56	64	51	75	28	9	37	6
Tarefa 6	59	65	85	40	23	39	99	46	17	94	6	67	69	86	8
Tarefa 7	10	7	22	36	31	75	57	49	44	21	77	70	64	46	69
Tarefa 8	53	74	93	26	54	89	82	66	37	63	71	17	58	4	46
Tarefa 9	76	72	42	17	27	56	78	5	72	19	90	46	43	56	17
Tarefa 10	18	79	93	71	48	23	20	90	94	87	6	36	84	25	83
Tarefa 11	52	61	45	60	15	74	49	26	94	54	1	58	56	54	72
Tarefa 12	63	73	82	84	15	54	52	52	36	21	45	41	21	97	50
Tarefa 13	90	90	77	33	31	26	14	75	92	70	55	56	39	49	23
Tarefa 14	87	47	58	34	29	83	24	48	97	89	84	82	53	99	10
Tarefa 15	35	32	30	93	58	28	88	16	98	4	82	98	26	29	77
Tarefa 16	18	92	62	59	3	94	34	56	24	18	66	53	30	41	10
Tarefa 17	2	26	17	18	60	39	23	95	81	56	34	8	47	72	56
Tarefa 18	6	79	65	58	94	45	80	3	29	80	27	60	94	14	76
Tarefa 19	31	79	87	79	57	48	33	42	93	86	54	32	8	16	63
Tarefa 20	96	1	75	42	45	51	10	58	71	92	23	18	63	27	63
Tarefa 21	84	82	16	61	43	75	28	15	19	93	22	1	62	9	5
Tarefa 22	46	29	50	12	72	18	79	73	23	1	58	1	95	25	71
Tarefa 23	10	39	49	56	71	40	90	28	89	42	9	92	52	6	20
Tarefa 24	70	63	68	97	86	81	38	7	53	48	43	59	88	29	87
Tarefa 25	81	97	65	60	15	29	9	80	78	85	95	85	91	28	92
Tarefa 26	39	6	59	34	34	32	12	7	35	4	53	69	89	3	40
Tarefa 27	98	85	51	9	24	7	59	98	50	98	64	31	31	29	1
Tarefa 28	59	68	3	8	2	9	69	14	72	84	69	54	45	59	7
Tarefa 29	92	21	53	64	59	79	52	14	61	86	82	98	83	24	87
Tarefa 30	51	70	94	80	35	56	8	94	11	3	60	73	26	21	45

Tabela A.4 Roteiros de máquinas do cenário de *job shop* 50 tarefas x 20 máquinas(ta61).

	Máquina responsável pela etapa da tarefa																			
Tarefa 1	9	20	18	16	2	3	11	8	14	19	6	7	4	5	15	12	17	1	10	13
Tarefa 2	13	12	4	17	2	3	10	9	19	5	1	20	18	16	6	11	14	15	7	8
Tarefa 3	9	7	16	15	4	8	20	19	2	12	5	13	18	14	1	3	10	11	17	6
Tarefa 4	13	10	6	17	1	3	14	19	7	12	2	20	9	15	8	5	16	11	4	18
Tarefa 5	2	4	14	5	19	6	7	10	8	9	16	1	15	11	13	20	17	18	3	12
Tarefa 6	17	4	5	15	2	3	11	10	6	1	19	20	9	18	13	16	14	7	8	12
Tarefa 7	20	5	16	1	13	14	2	8	19	6	10	4	17	9	12	3	7	15	11	18
Tarefa 8	10	11	9	5	15	6	16	8	19	7	14	20	18	3	17	12	1	13	4	2
Tarefa 9	10	14	16	8	19	3	12	20	2	9	1	5	4	13	7	17	15	18	11	6
Tarefa 10	7	12	18	9	17	8	1	15	5	16	3	14	19	2	10	20	11	6	4	13
Tarefa 11	14	5	8	7	16	12	11	10	4	18	17	6	13	2	9	19	15	1	20	3
Tarefa 12	4	11	15	1	8	13	10	20	12	6	19	2	14	18	3	7	5	17	16	9
Tarefa 13	1	11	3	19	15	13	14	2	4	10	18	16	20	5	6	9	17	7	8	12
Tarefa 14	7	12	10	9	16	18	1	5	15	11	13	3	14	19	6	2	17	20	4	8
Tarefa 15	2	10	17	18	16	3	6	11	9	15	8	12	14	1	20	4	5	19	13	7
Tarefa 16	12	18	6	2	13	19	11	10	3	16	17	7	8	9	15	5	1	20	14	4
Tarefa 17	4	6	14	17	15	12	18	13	8	19	16	9	3	1	10	11	5	7	20	2
Tarefa 18	17	14	9	18	6	11	19	13	2	5	15	10	7	12	4	8	16	3	1	20
Tarefa 19	5	15	16	20	18	19	2	10	9	8	7	6	1	4	17	12	11	13	14	3
Tarefa 20	10	16	14	5	15	9	4	3	6	19	8	17	7	12	11	1	18	13	20	2
Tarefa 21	19	3	18	17	11	9	13	20	15	2	7	6	1	8	12	4	5	16	14	10
Tarefa 22	9	18	12	16	4	7	17	14	13	3	20	10	15	5	19	2	8	11	6	1
Tarefa 23	12	18	11	20	13	19	17	6	16	5	2	14	3	8	1	7	15	10	4	9
Tarefa 24	19	8	16	10	4	20	17	12	6	14	18	11	13	1	2	15	3	5	7	9
Tarefa 25	5	18	7	3	19	6	13	8	1	15	12	16	2	20	17	9	4	10	14	11
Tarefa 26	7	2	16	3	19	12	8	15	11	5	1	4	10	14	13	20	18	9	6	17
Tarefa 27	16	7	19	18	1	20	2	15	17	10	14	5	6	9	3	13	12	4	8	11
Tarefa 28	10	18	11	7	4	1	13	19	17	8	15	5	14	2	9	20	12	6	3	16
Tarefa 29	14	5	18	15	6	12	9	1	10	19	13	17	3	7	16	4	11	2	20	8
Tarefa 30	3	5	11	13	18	6	8	12	17	16	4	10	1	19	20	7	2	14	9	15
Tarefa 31	10	3	16	8	12	18	15	7	13	4	20	1	11	2	14	9	17	6	5	19
Tarefa 32	18	5	20	16	1	4	7	3	13	14	2	9	17	8	6	11	15	19	10	12
Tarefa 33	14	7	5	13	6	11	12	9	3	19	10	4	15	1	18	2	16	17	20	8
Tarefa 34	16	3	4	8	14	17	13	15	9	5	6	1	18	11	7	19	10	12	20	2
Tarefa 35	16	9	11	2	19	12	1	18	5	13	4	20	6	3	15	10	17	8	7	14
Tarefa 36	5	13	4	19	11	20	18	7	3	2	1	6	8	9	12	15	10	17	14	16
Tarefa 37	17	18	10	20	8	2	7	9	4	14	16	13	12	3	15	6	19	5	1	11
Tarefa 38	13	3	1	14	5	17	8	4	19	16	6	10	7	11	12	2	20	15	18	9
Tarefa 39	16	2	3	17	13	20	1	9	10	6	8	12	19	5	11	4	14	15	7	18
Tarefa 40	8	1	13	11	2	19	9	6	20	4	12	16	3	17	14	7	18	5	10	15
Tarefa 41	12	3	11	18	8	5	7	20	17	9	10	1	2	6	13	16	14	19	15	4
Tarefa 42	16	9	8	6	20	18	19	15	7	17	4	3	11	10	14	13	1	12	5	2
Tarefa 43	5	4	13	20	8	12	16	1	15	10	11	19	14	6	3	7	9	18	17	2
Tarefa 44	3	8	5	19	18	2	6	9	14	10	16	4	13	7	12	17	1	20	15	11
Tarefa 45	14	15	2	3	17	1	12	20	7	8	6	11	16	13	18	10	5	19	4	9
Tarefa 46	1	17	4	8	3	18	20	5	13	9	6	7	15	10	19	11	14	2	12	16
Tarefa 47	7	14	13	10	3	20	15	19	5	6	11	8	16	9	4	2	1	12	17	18
Tarefa 48	9	15	8	11	10	7	1	3	17	13	14	6	20	18	5	16	4	12	19	2
Tarefa 49	4	11	8	1	2	14	19	18	9	15	17	16	12	20	6	13	5	10	7	3
Tarefa 50	16	6	19	20	18	10	13	15	2	12	3	8	11	1	5	7	17	9	4	14

Tabela A.5 Tempos de operações do cenário de *job shop* 50 tarefas x 20 máquinas(ta61).

	Tempo de duração de cada etapa da tarefa																			
Tarefa 1	48	40	54	71	52	70	41	76	52	24	5	43	68	10	49	9	81	30	93	17
Tarefa 2	85	18	54	42	41	71	68	82	54	49	21	1	58	1	69	58	40	59	66	29
Tarefa 3	33	34	77	42	95	2	71	73	19	25	45	88	19	40	42	17	81	72	70	67
Tarefa 4	51	41	74	97	26	4	25	12	17	76	6	79	49	39	1	27	44	75	1	18
Tarefa 5	22	99	7	7	72	24	19	81	23	72	50	95	31	67	67	22	12	28	68	88
Tarefa 6	52	51	44	38	64	11	62	20	54	15	83	79	55	48	38	37	42	81	89	60
Tarefa 7	82	43	57	1	89	11	41	50	68	2	4	65	20	56	46	36	33	56	13	50
Tarefa 8	45	11	63	59	69	39	44	61	67	72	74	59	16	26	90	66	56	47	95	39
Tarefa 9	92	2	88	90	45	88	90	94	34	1	81	64	70	55	7	33	21	35	62	61
Tarefa 10	89	21	61	18	77	20	42	59	79	12	56	14	21	43	89	31	71	92	47	71
Tarefa 11	61	84	3	73	35	36	79	88	54	96	22	70	10	4	76	40	85	84	93	65
Tarefa 12	68	72	74	97	63	33	96	4	63	31	1	98	39	65	72	20	7	63	33	26
Tarefa 13	41	65	34	71	19	49	87	61	79	61	29	22	74	68	60	23	82	33	94	42
Tarefa 14	17	40	40	28	6	62	83	95	44	91	79	39	68	79	1	20	96	62	62	70
Tarefa 15	39	89	37	7	84	60	61	73	64	73	3	75	3	48	74	67	39	32	69	25
Tarefa 16	9	83	30	3	31	93	86	49	34	91	56	80	33	77	35	63	72	46	22	73
Tarefa 17	21	46	33	54	22	64	20	76	77	97	28	54	81	95	81	72	80	75	18	81
Tarefa 18	52	30	38	70	22	15	66	26	55	34	13	65	87	38	85	89	77	22	67	44
Tarefa 19	63	95	18	94	73	51	35	57	38	65	69	60	90	68	32	40	11	75	97	51
Tarefa 20	68	37	39	13	76	77	6	6	53	41	72	71	46	24	46	50	12	39	92	54
Tarefa 21	93	95	8	27	53	75	3	42	5	24	73	88	57	20	99	39	74	75	44	24
Tarefa 22	83	14	66	96	11	36	20	5	72	38	79	10	27	27	90	8	83	10	61	69
Tarefa 23	22	56	54	50	51	9	15	36	20	79	51	84	40	59	48	27	65	44	40	83
Tarefa 24	5	75	43	17	10	92	22	36	7	71	77	70	10	24	78	77	56	42	16	48
Tarefa 25	37	96	81	12	92	86	63	88	28	57	58	23	4	95	80	12	82	53	5	75
Tarefa 26	58	59	65	78	68	50	38	97	72	94	59	42	5	19	27	54	69	2	56	51
Tarefa 27	4	7	36	35	80	95	51	59	93	5	61	4	43	30	93	76	42	99	30	46
Tarefa 28	88	75	81	40	61	94	78	24	19	44	96	23	90	94	80	97	24	44	54	52
Tarefa 29	5	99	60	87	64	36	78	32	4	18	26	87	74	26	90	45	35	54	27	23
Tarefa 30	93	95	11	14	99	86	41	26	50	74	21	6	67	87	46	84	11	89	89	66
Tarefa 31	50	71	71	5	60	29	17	29	98	61	87	58	6	60	84	92	23	25	23	57
Tarefa 32	75	60	77	48	87	52	98	8	55	97	55	68	59	90	50	98	57	43	72	35
Tarefa 33	46	22	11	49	34	30	79	72	77	47	55	63	58	89	71	94	95	13	97	46
Tarefa 34	25	98	71	68	8	72	57	39	83	17	90	31	81	6	97	98	82	82	52	82
Tarefa 35	42	77	71	19	80	31	66	90	18	15	76	58	92	34	66	8	65	67	84	42
Tarefa 36	41	42	69	81	95	16	45	52	48	35	72	80	81	4	3	4	96	53	14	80
Tarefa 37	6	6	12	86	26	52	70	93	81	31	89	99	99	71	74	7	43	86	1	93
Tarefa 38	44	54	36	40	68	49	45	58	44	65	72	65	53	48	90	98	60	71	27	48
Tarefa 39	9	16	56	27	50	57	55	87	44	47	29	82	80	43	75	10	70	38	28	2
Tarefa 40	29	91	85	51	86	34	73	12	14	51	1	38	74	92	60	43	36	23	82	30
Tarefa 41	21	97	4	85	21	55	34	62	78	11	34	17	3	43	38	44	45	17	3	83
Tarefa 42	29	6	45	15	60	29	97	91	13	8	50	46	72	86	7	30	28	13	27	42
Tarefa 43	38	10	93	6	72	38	73	88	44	66	79	47	61	6	64	18	2	6	91	37
Tarefa 44	21	20	51	96	51	42	52	37	85	18	44	60	68	3	6	20	81	96	30	9
Tarefa 45	16	54	53	57	46	84	1	76	26	7	69	88	29	73	32	51	4	74	75	75
Tarefa 46	27	54	90	25	97	68	14	54	29	14	8	1	60	13	16	41	81	35	18	79
Tarefa 47	56	7	31	55	85	35	82	63	35	54	52	77	82	94	81	25	24	56	23	79
Tarefa 48	33	50	22	70	59	51	80	84	47	88	27	18	34	47	4	41	56	42	26	66
Tarefa 49	31	83	9	34	62	83	61	41	58	96	87	18	56	2	95	21	51	13	31	96
Tarefa 50	62	95	8	3	27	19	36	97	87	62	86	21	37	11	11	67	84	34	48	97

Apêndice B

NOÇÕES DE ESTATÍSTICA

Todos os conceitos apresentados neste apêndice vieram das seguintes referências: (MONTGOMERY, 2005; MONTGOMERY, 2009; MORETTIN; BUSSAB, 2004; MCCLAVE, 2009; TRIOLA, 2008; ANDERSON, 2002).

O tamanho amostral, testes de normalidade das distribuições e testes de hipóteses possuem várias formas de serem calculados e, usualmente, lança-se mão de ferramentas computacionais.

B.1 Testes de Hipóteses *t-student*

Nesta seção, pretende-se explicar os conceitos básicos para o entendimento do uso de *t-student* para teste de hipóteses, fazendo inferência na diferença das médias de duas amostras independentes com distribuições normais e com variâncias desconhecidas.

Fórmulas complexas e explicações mais detalhadas não fazem parte do contexto desta obra. Limitou-se aos conceitos e forma de interpretação de seus resultados, uma vez que todos os métodos comentados possuem implementação em vários programas de distribuição gratuita.

Amostras independentes são duas amostras que não possuem algum tipo de correlação entre elas.

Variância é a raiz quadrada do desvio padrão.

Um teste de hipóteses formula duas hipóteses: a hipótese nula (HN) e a hipótese alternativa (HA). Um teste de hipóteses tenta aceitar ou rejeitar a hipótese nula e, pela sua rejeição, aceitar a hipótese alternativa.

Para garantir qualidade das inferências, devem-se considerar os dois possíveis erros:

- Erro I – Quando se rejeita a hipótese nula e esta é verdadeira. Neste caso, a sua probabilidade de ocorrência chama-se nível de significância (α). Quanto menor

esse nível, maior a confiança em não se rejeitar a hipótese nula erroneamente. Este nível, normalmente, é estipulado com valores de 0,05 a 0,001. Confiança do teste é dado por (1 – α);

- Erro II – Quando não se rejeita a hipótese nula e esta é falsa. Neste caso, a sua probabilidade de ocorrer é denotada por β. Chama-se poder do teste ao valor (1 – β). β é importante na definição do tamanho amostral.

Para o teste de hipótese usando *t-student* algumas condições devem ser atendidas:

- Definir nível de significância (α), definido nesta tese como 0,05;
- Definir poder do teste (1-β), definido nesta tese como 0,95. Caso nesse poder de teste o tamanho amostral extrapole 200, estabelece-se o tamanho amostral como 200 e calcula-se o poder de teste para esse tamanho amostral;
- Garantir que as amostras: sejam grandes (>30); ou que possuem distribuição normal suficientemente boa para que se possa inferir, verificado com o uso dos métodos indicados na próxima seção; ou pela observação do papel de probabilidade;
- Garantir que as amostras são independentes, o que, normalmente, ocorre com amostras geradas aleatoriamente.

A hipótese nula é rejeitada sempre que o valor de p (p-valor) for menor que o nível de significância. Quando a hipótese nula é rejeitada, automaticamente se aceita a hipótese alternativa.

Os pares HN e HA normalmente formuladas são:

1 - HN afirma que a tese possui resultados iguais aos do concorrente e HA afirma que a tese possui resultados diferentes aos do concorrente;

2 - HN afirma que a tese possui resultados menores ou iguais aos do concorrente e HA afirma que a tese possui resultados maiores que os do concorrente;

3 - HN afirma que a tese possui resultados maiores ou iguais aos do concorrente e HA afirma que a tese possui resultados menores que os do concorrente;

Para verificar-se se a hipótese possui resultados menores que os do concorrente, faz-se necessário elaborar dois pares de HN e HA. O primeiro com HN afirmando que a hipótese possui valores menores ou iguais aos do concorrente e o segundo com HN afirmando que a

hipótese possui resultados iguais aos do concorrente. A aceitação da primeira HN e a rejeição da segunda HN implica em afirmar que a hipótese possui resultados menores que os do concorrente.

Para verificar a veracidade do parágrafo acima segue uma prova usando lógica proposicional.

Sejam as proposições:
p: a hipótese possui valores menores que o concorrente;
q: a hipótese possui valores iguais ao concorrente;
HN1: a hipótese possui valores menores ou iguais ao concorrente (p ou q);
HN2: a hipótese possui valores iguais ao concorrente (q).
Se HN1 não for rejeitada então p ou q = V;
Se HN2 for rejeitada então q = F;
Substituindo q em HN1 pelo seu valor verdade então p ou F = V;
Usando a propriedade de identidade então p = V.

Os passos para se efetuar uma comparação de duas amostras usando testes de hipótese com *t-student* são os seguintes:

- Formular as hipóteses nula (HN) e alternativa (HA);
- Definir o nível de significância (quanto se permite rejeitar HN erroneamente);
- Definir o poder do teste (quanto se permite aceitar a HN erroneamente);
- Gerar duas amostras de tamanho razoável para poder calcular os tamanhos amostrais. Considera-se que estas amostras são independentes, com distribuições normais e variâncias desconhecidas;
- Calcular o tamanho amostral usando como entradas os parâmetros indicados na Seção B.3;
- Gerar os complementos das amostras, se necessário;
- Testar a suposição de normalidade das amostras, conforme indicado na Seção B.2;
- Aplicar o teste de hipóteses;
- Se o p-valor obtido for superior ao grau de significância do teste, então a hipótese é aceita (corroborada), caso o p-valor seja inferior ao grau de significância do teste, então a hipótese é rejeitada.

B.2 Método de verificação de normalidade de uma distribuição

Considera-se que uma amostra possui distribuição normal quando:

- Possuir mais de 30 elementos (Teorema do Limite Central);
- For testada por métodos específicos com esse fim (Anderson-Darling, Kolmogorov-Smirnov e Shapiro-Wilk) e for aceita. Sempre se aceita quando p-valor for maior ou igual a 0,05. No processo destes testes pode ocorrer de serem rejeitados, sendo possível então partir para duas estratégias de manipulação de população antes de refazer o cálculo:
    - 1 – elevar toda a população a alguma potência;
    - 2 – aumentar o tamanho das populações, quando possível.

    Estes testes normalmente são usados quando o número de elementos da amostra for inferior a 30 ou quando há fortes indícios de que a distribuição não seja normal.

- Pode-se perceber sua normalidade pela sua curva de distribuição de frequência de ocorrências de seus valores, a qual se assemelha a uma curva de Gauss;
- Analisando seu papel de probabilidade da amostra. Papel de probabilidade é um gráfico que plota uma diagonal e os pontos referentes às amostras, passados por uma função. Considera-se distribuição normal quando estes pontos não fogem muito dessa diagonal. Percebe-se que é um teste subjetivo, mas comumente usado.

  Estas duas últimas formas são bastante subjetivas e normalmente são usadas mais como forma de ilustrar a escolha.

B.3 Método de cálculo de tamanho amostral por *t-student*

Um fator que influencia a qualidade do poder de teste é o tamanho das amostras usado nele. O tamanho amostral é calculado usando-se as médias, desvios padrões e tamanho do efeito das amostras iniciais. Este tamanho indica o tamanho mínimo que a amostra deve possuir para que se obtenha o poder de teste desejado.

Tamanho do efeito é calculado da seguinte forma: divide-se o valor absoluto da diferença das médias das amostras pela média de seus desvios padrões.

$$te = |u_1 - u_2|/((\sigma_1 + \sigma_2)/2),$$

Onde te é o tamanho de efeito, $u_1\ e\ u_2$ são as médias das amostras e $\sigma_1\ e\ \sigma_2$ são os desvios padrões das amostras.

Como o cálculo do tamanho amostral é feito de forma recursiva, a maioria dos pacotes estatísticos gratuitos o possuem implementado.

Apêndice C

RESULTADOS DA EXECUÇÃO DO PLANEJAMENTO DE EXPERIMENTOS

Neste apêndice, são apresentados os cálculos e tabelas dos dados, usados como suporte para o capítulo de validação.

Todos os cálculos são feitos usando o Excel e o suplemento Action (PORTAL ACTION, 2015), criado usando a linguagem R e de distribuição gratuita.

C.1 Amostras iniciais de cada trabalho em cada cenário

Nesta seção, apresentam-se as populações amostrais iniciais de 35 execuções, para os quatro trabalhos em cada um dos três cenários.

A Tabela C.1 apresenta as populações iniciais dos quatro trabalhos no cenário 15x15 (ta01), a Tabela C.2 apresenta as populações iniciais dos quatro trabalhos no cenário 30x15 (ta31) e a Tabela C.3 apresenta as populações iniciais dos quatro trabalhos no cenário 50x20 (ta61).

Nas três tabelas, a coluna 1 indica a execução a que se referem os dados e as colunas de 2 a 5 apresentam os melhores *makespans* obtidos pelos quatro trabalhos neste cenário ao final do tempo limite de cinco minutos. As duas linhas finais de cada tabela indicam as médias e os desvios padrões dos *makespans* para cada trabalho.

Tabela C.1 Amostra inicial de dados para o cenário 15x15 para os quatro trabalhos.

Execução	TBACO	TGLOBAL	TRN	TCC
1	1327	1323	1331	1331
2	1316	1339	1351	1320
3	1300	1316	1325	1319
4	1319	1334	1326	1323
5	1300	1366	1350	1313
6	1329	1350	1326	1318
7	1316	1362	1313	1326
8	1329	1349	1319	1326
9	1319	1340	1300	1315
10	1323	1349	1325	1336
11	1300	1332	1328	1323
12	1323	1330	1320	1334
13	1321	1341	1318	1333
14	1306	1343	1321	1306
15	1300	1320	1319	1311
16	1323	1332	1332	1306
17	1342	1312	1337	1331
18	1314	1325	1340	1320
19	1321	1326	1323	1340
20	1300	1320	1323	1320
21	1323	1324	1326	1326
22	1332	1323	1321	1317
23	1316	1321	1310	1343
24	1335	1309	1319	1340
25	1310	1331	1336	1322
26	1333	1315	1340	1317
27	1326	1338	1323	1326
28	1324	1306	1323	1322
29	1316	1334	1311	1344
30	1325	1322	1324	1335
31	1315	1323	1314	1329
32	1300	1366	1319	1326
33	1310	1323	1323	1339
34	1319	1345	1326	1323
35	1325	1340	1334	1331
Média	1318,20	1332,26	1325,03	1325,46
Desvio padrão	11,09	15,19	10,54	9,85

Tabela C.2 Amostra inicial de dados para o cenário 30x15 para os quatro trabalhos.

Execução	TBACO	TGLOBAL	TRN	TCC
1	1953	1977	1966	1969
2	1946	1965	1969	1958
3	1930	1939	1938	1924
4	1955	1985	1949	1963
5	1962	1986	1931	2002
6	1948	1980	1972	1956
7	1955	2007	1994	1956
8	1947	1966	1975	1940
9	1957	1955	1948	1967
10	1969	2006	1985	1958
11	1955	2013	1995	1921
12	1980	1979	1969	1975
13	1978	1985	1993	1974
14	1975	2006	1991	1979
15	1942	1969	1914	1937
16	1948	2000	1938	1971
17	1963	1984	1965	1986
18	1954	1974	1983	1977
19	1960	1961	1976	1964
20	1965	1971	1990	1973
21	1946	2008	1949	1991
22	1969	2019	1950	1975
23	1953	2015	1950	1962
24	1955	1993	1982	1986
25	1965	2033	1966	1941
26	1961	1977	1955	1988
27	1945	1997	1938	1923
28	1961	1958	1952	1927
29	1965	1971	1975	1984
30	1933	1976	1951	1951
31	1957	1947	1984	1992
32	1955	1988	1984	1944
33	1934	1988	1973	1979
34	1968	1998	1982	1964
35	1981	2009	1979	1982
Média	1956,86	1985,28	1966,03	1963,97
Desvio padrão	12,53	21,56	20,39	21,32

Tabela C.3 Amostra inicial de dados para o cenário 50x20 para os quatro trabalhos.

Execução	TBACO	TGLOBAL	TRN	TCC
1	3258	3293	3227	3347
2	3232	3254	3256	3253
3	3199	3258	3282	3237
4	3212	3255	3261	3283
5	3196	3301	3251	3246
6	3238	3288	3234	3254
7	3269	3272	3231	3290
8	3250	3205	3218	3300
9	3225	3261	3226	3235
10	3241	3234	3271	3275
11	3279	3259	3236	3330
12	3222	3271	3214	3265
13	3220	3252	3212	3288
14	3198	3228	3266	3299
15	3265	3294	3309	3360
16	3258	3329	3240	3292
17	3256	3314	3264	3268
18	3227	3261	3238	3321
19	3228	3245	3240	3245
20	3208	3271	3245	3250
21	3240	3276	3261	3331
22	3200	3215	3241	3273
23	3203	3247	3214	3294
24	3218	3287	3265	3261
25	3232	3276	3209	3257
26	3260	3233	3246	3224
27	3261	3327	3267	3295
28	3243	3222	3253	3277
29	3287	3227	3310	3324
30	3266	3272	3256	3282
31	3271	3241	3294	3269
32	3242	3318	3300	3342
33	3226	3250	3271	3257
34	3276	3250	3256	3330
35	3204	3252	3253	3265
Média	3237,43	3263,94	3251,91	3283,40
Desvio padrão	26,17	30,98	26,43	34,71

C.2 Cálculo dos tamanhos amostrais

Nesta seção, são apresentados os cálculos amostrais para cada par em comparação, sempre sendo a TBACO e um dos trabalhos de comparação.

Os parâmetros são os seguintes:

Nível de significância = 0,05

Poder do teste = 0,95

Cada tabela apresenta os dados dos cálculos amostrais para cada par de comparações em cada cenário. Para cada par de trabalhos estes cálculos são feitos com as 35 amostras apresentadas na seção anterior. Para cada par do cálculo amostral mostram-se as médias e os desvios padrões de ambos os trabalhos, a diferença entre essas médias, a média dos desvios padrões das duas amostras, o tamanho do efeito calculado e o tamanho amostral sugerido.

Tabela C.4 Cálculos dos tamanhos amostrais para comparação da proposta TBACO com os trabalhos TGLOBAL, TRN e TCC no cenário 15x15.

Número de amostras	Trabalhos				Desvio padrão médio	Diferença das médias	Efeito de tamanho	Tamanho amostral calculado
	TBACO		TGLOBAL					
	Média	Desvio padrão	Média	Desvio padrão				
35	1318,20	11,09	1332,26	15,19	13,14	14,06	1,07	24
Número de amostras	**Trabalhos**				**Desvio padrão médio**	**Diferença das médias**	**Efeito de tamanho**	**Tamanho amostral calculado**
	TBACO		**TRN**					
	Média	**Desvio padrão**	**Média**	**Desvio padrão**				
35	1316,43	12,31	1324,84	11,31	11,81	8,41	0,71	67
Número de amostras	**Trabalhos**				**Desvio padrão médio**	**Diferença das médias**	**Efeito de tamanho**	**Tamanho amostral calculado**
	TBACO		**TCC**					
	Média	**Desvio padrão**	**Média**	**Desvio padrão**				
35	1316,48	12,17	1323,72	9,93	11,05	7,23	0,65	56

Tabela C.5 Cálculos dos tamanhos amostrais para comparação da proposta TBACO com os trabalhos TGLOBAL, TRN e TCC no cenário 30x15.

Número de amostras	Trabalhos				Desvio padrão médio	Diferença das médias	Efeito de tamanho	Tamanho amostral calculado
	TBACO		TGLOBAL					
	Média	Desvio padrão	Média	Desvio padrão				
35	1956,86	12,53	1985,29	21,56	17,04	28,43	1,67	11
Número de amostras	Trabalhos				Desvio padrão médio	Diferença das médias	Efeito de tamanho	Tamanho amostral calculado
	TBACO		TRN					
	Média	Desvio padrão	Média	Desvio padrão				
35	1955,36	14,50	1970,10	19,68	17,09	14,74	0,86	85
Número de amostras	Trabalhos				Desvio padrão médio	Diferença das médias	Efeito de tamanho	Tamanho amostral calculado
	TBACO		TCC					
	Média	Desvio padrão	Média	Desvio padrão				
35	1954,52	15,50	1963,82	20,14	17,82	9,30	0,52	149

Tabela C.6 Cálculos dos tamanhos amostrais para comparação da proposta TBACO com os trabalhos TGLOBAL, TRN e TCC no cenário 50x20.

Número de amostras	Trabalhos				Desvio padrão médio	Diferença das médias	Efeito de tamanho	Tamanho amostral calculado
	TBACO		TGLOBAL					
	Média	Desvio padrão	Média	Desvio padrão				
35	3237,43	26,17	3263,94	30,98	28,57	26,51	0,93	32
Número de amostras	Trabalhos				Desvio padrão médio	Diferença das médias	Efeito de tamanho	Tamanho amostral calculado
	TBACO		TRN					
	Média	Desvio padrão	Média	Desvio padrão				
35	3236,14	24,44	3248,77	27,96	25,85	12,62	0,49	87
Número de amostras	Trabalhos				Desvio padrão médio	Diferença das médias	Efeito de tamanho	Tamanho amostral calculado
	TBACO		TCC					
	Média	Desvio padrão	Média	Desvio padrão				
35	3237,43	26,17	3283,40	34,71	30,44	45,97	1,51	13

A Tabela C.7 resume os tamanhos amostrais para cada par de trabalhos em cada cenário. Note-se que a TBACO sempre assume o maior dos três valores de tamanhos amostrais calculados.

Tabela C.7 Resumo dos tamanhos amostrais criados para cada trabalho em cada cenário.

Cenário	Trabalho	Tamanho amostral usado
Cenário 15 X 15	TBACO	70
	TGLOBAL	35
	TRN	70
	TCC	60
Cenário 30 X 15	TBACO	150
	TGLOBAL	35
	TRN	90
	TCC	150
Cenário 50 X 20	TBACO	90
	TGLOBAL	35
	TRN	90
	TCC	35

C.3 Populações de amostras completas

Durante o processo de cálculo do tamanho amostral, apresentado na seção anterior, criam-se as amostras complementares às que já existiam, completando o número de amostras necessário para que o poder de teste fosse de 95%.

As tabelas a seguir indicam estas novas amostras. Note que os dados da TBACO sempre são calculados com base no maior tamanho amostral calculado para os trabalhos a serem com ele comparados.

As tabelas estão desenhadas em cinco conjuntos de dados, com cada par de colunas representando um quinto do total, de forma a ser possível colocar, no mesmo tipo de tabela, populações amostrais pequenas e grandes, sem extrapolar o limite de uma página.

C.3.1 Amostras Totais para o cenário 15x15

Tabela C.8 Amostras finais de dados para o trabalho TGLOBAL no cenário 15x15.

Execução	Makespan	Execução	Makespan	Execução	Makespan	Execução	Makespan	Execução	Makespan
1	1323	8	1349	15	1320	22	1323	29	1334
2	1339	9	1340	16	1332	23	1321	30	1322
3	1316	10	1349	17	1312	24	1309	31	1323
4	1334	11	1332	18	1325	25	1331	32	1366
5	1366	12	1330	19	1326	26	1315	33	1323
6	1350	13	1341	20	1320	27	1338	34	1345
7	1362	14	1343	21	1324	28	1306	35	1340

Tabela C.9 Amostras finais de dados para o trabalho TRN no cenário 15x15.

Execução	*Makespan*	Execução	*Makespan*	Execução	*Makespan*	Execução	*Makespan*	Execução	*Makespan*
1	1331	15	1319	29	1311	43	1321	57	1319
2	1351	16	1332	30	1324	44	1327	58	1303
3	1325	17	1337	31	1314	45	1327	59	1347
4	1326	18	1340	32	1319	46	1315	60	1329
5	1350	19	1323	33	1323	47	1329	61	1313
6	1326	20	1323	34	1326	48	1300	62	1325
7	1313	21	1326	35	1334	49	1335	63	1332
8	1319	22	1321	36	1322	50	1320	64	1323
9	1300	23	1310	37	1316	51	1319	65	1344
10	1325	24	1319	38	1321	52	1332	66	1311
11	1328	25	1336	39	1320	53	1346	67	1340
12	1320	26	1340	40	1331	54	1335	68	1331
13	1318	27	1323	41	1311	55	1327	69	1298
14	1321	28	1323	42	1326	56	1345	70	1323

Tabela C.10 Amostras finais de dados para o trabalho TCC no cenário 15x15.

Execução	*Makespan*	Execução	*Makespan*	Execução	*Makespan*	Execução	*Makespan*	Execução	*Makespan*
1	1331	13	1333	25	1322	37	1315	49	1323
2	1320	14	1306	26	1317	38	1326	50	1312
3	1319	15	1311	27	1326	39	1328	51	1300
4	1323	16	1306	28	1322	40	1336	52	1325
5	1313	17	1331	29	1344	41	1319	53	1303
6	1318	18	1320	30	1335	42	1304	54	1328
7	1326	19	1340	31	1329	43	1304	55	1327
8	1326	20	1320	32	1326	44	1324	56	1322
9	1315	21	1326	33	1339	45	1322	57	1327
10	1336	22	1317	34	1323	46	1332	58	1334
11	1323	23	1343	35	1331	47	1325	59	1323
12	1334	24	1340	36	1323	48	1324	60	1326

Tabela C.11 Amostras finais de dados para o trabalho TBACO no cenário 15x15.

Execução	*Makespan*	Execução	*Makespan*	Execução	*Makespan*	Execução	*Makespan*	Execução	*Makespan*
1	1327	15	1300	29	1316	43	1318	57	1309
2	1316	16	1323	30	1325	44	1331	58	1327
3	1300	17	1342	31	1315	45	1304	59	1310
4	1319	18	1314	32	1300	46	1324	60	1300
5	1300	19	1321	33	1310	47	1332	61	1313
6	1329	20	1300	34	1319	48	1296	62	1342
7	1316	21	1323	35	1325	49	1330	63	1322
8	1329	22	1332	36	1300	50	1327	64	1316
9	1319	23	1316	37	1300	51	1338	65	1324
10	1323	24	1335	38	1318	52	1300	66	1315
11	1300	25	1310	39	1309	53	1308	67	1329
12	1323	26	1333	40	1305	54	1310	68	1300
13	1321	27	1326	41	1338	55	1318	69	1300
14	1306	28	1324	42	1300	56	1300	70	1300

C.3.2 Amostras Totais para o cenário 30x15

Tabela C.12 Amostras finais de dados para o trabalho TGLOBAL no cenário 30x15.

Execução	*Makespan*	Execução	*Makespan*	Execução	*Makespan*	Execução	*Makespan*	Execução	*Makespan*
1	1977	8	1966	15	1969	22	2019	29	1971
2	1965	9	1955	16	2000	23	2015	30	1976
3	1939	10	2006	17	1984	24	1993	31	1947
4	1985	11	2013	18	1974	25	2033	32	1988
5	1986	12	1979	19	1961	26	1977	33	1988
6	1980	13	1985	20	1971	27	1997	34	1998
7	2007	14	2006	21	2008	28	1958	35	2009

Tabela C.13 Amostras finais de dados para o trabalho TRN no cenário 30x15.

Execução	*Makespan*	Execução	*Makespan*	Execução	*Makespan*	Execução	*Makespan*	Execução	*Makespan*
1	1966	19	1976	37	1966	55	1967	73	1999
2	1969	20	1990	38	1989	56	1984	74	1949
3	1938	21	1949	39	1976	57	1979	75	1992
4	1949	22	1950	40	1946	58	1973	76	1983
5	1931	23	1950	41	1969	59	1975	77	1975
6	1972	24	1982	42	1974	60	1963	78	1966
7	1994	25	1966	43	1984	61	1994	79	1974
8	1975	26	1955	44	1978	62	1997	80	1974
9	1948	27	1938	45	1950	63	1973	81	1949
10	1985	28	1952	46	1977	64	1952	82	1961
11	1995	29	1975	47	1988	65	1961	83	1989
12	1969	30	1951	48	1943	66	1999	84	1967
13	1993	31	1984	49	1984	67	1910	85	1954
14	1991	32	1984	50	1993	68	1964	86	1970
15	1914	33	1973	51	1932	69	1992	87	1976
16	1938	34	1982	52	2003	70	1977	88	1970
17	1965	35	1979	53	2010	71	1969	89	1977
18	1983	36	1978	54	1940	72	1965	90	1999

Tabela C.14 Amostras finais de dados para o trabalho TCC no cenário 30x15.

Execução	*Makespan*	Execução	*Makespan*	Execução	*Makespan*	Execução	*Makespan*	Execução	*Makespan*
1	1969	31	1992	61	1974	91	1972	121	1968
2	1958	32	1944	62	1951	92	1952	122	1940
3	1924	33	1979	63	1982	93	1987	123	2001
4	1963	34	1964	64	1933	94	1954	124	1952
5	2002	35	1982	65	1974	95	1998	125	1918
6	1956	36	1957	66	1974	96	1959	126	1966
7	1956	37	1970	67	1941	97	1957	127	1968
8	1940	38	1973	68	1923	98	1950	128	1963
9	1967	39	1969	69	2004	99	1963	129	1949
10	1958	40	1963	70	1973	100	1969	130	1938
11	1921	41	1967	71	1961	101	1983	131	1955
12	1975	42	1957	72	1972	102	1956	132	1954
13	1974	43	1961	73	1944	103	1956	133	1966
14	1979	44	1936	74	1927	104	1949	134	1973
15	1937	45	1964	75	1945	105	1914	135	1963
16	1971	46	1974	76	1967	106	1959	136	1974
17	1986	47	1970	77	1953	107	1978	137	1951
18	1977	48	1976	78	1969	108	1952	138	1940
19	1964	49	2001	79	1978	109	2001	139	1973
20	1973	50	1999	80	2001	110	1958	140	1964
21	1991	51	1992	81	1951	111	1977	141	1989
22	1975	52	1978	82	1940	112	1984	142	1974
23	1962	53	1973	83	1967	113	1975	143	1966
24	1986	54	1964	84	1983	114	1971	144	1974
25	1941	55	1921	85	1942	115	1990	145	1941
26	1988	56	1949	86	1966	116	1969	146	2005
27	1923	57	1957	87	2026	117	1929	147	1935
28	1927	58	1966	88	1970	118	1970	148	1931
29	1984	59	1955	89	1970	119	1957	149	1987
30	1951	60	1960	90	1964	120	1952	150	1938

Tabela C.15 Amostras finais de dados para o trabalho TBACO no cenário 30x15.

Execução	*Makespan*	Execução	*Makespan*	Execução	*Makespan*	Execução	*Makespan*	Execução	*Makespan*
1	1953	31	1957	61	1949	91	1941	121	1948
2	1946	32	1955	62	1933	92	1947	122	1976
3	1930	33	1934	63	1963	93	1949	123	1969
4	1955	34	1968	64	1943	94	1957	124	1983
5	1962	35	1981	65	1963	95	1946	125	1933
6	1948	36	1941	66	1973	96	1974	126	1946
7	1955	37	1971	67	1939	97	1947	127	1909
8	1947	38	1954	68	1944	98	1948	128	1956
9	1957	39	1967	69	1967	99	1967	129	1931
10	1969	40	1949	70	1950	100	1955	130	1946
11	1955	41	1956	71	1968	101	1987	131	1959
12	1980	42	1986	72	1951	102	1955	132	1960
13	1978	43	1958	73	1947	103	1932	133	1939
14	1975	44	1933	74	1957	104	1954	134	1918
15	1942	45	1932	75	1965	105	1940	135	1940
16	1948	46	1983	76	1955	106	1957	136	1950
17	1963	47	1970	77	1961	107	1954	137	1980
18	1954	48	1961	78	1959	108	1962	138	1940
19	1960	49	1931	79	1973	109	1930	139	1954
20	1965	50	1971	80	1986	110	1933	140	1983
21	1946	51	1945	81	1940	111	1969	141	1973
22	1969	52	1940	82	1914	112	1947	142	1972
23	1953	53	1961	83	1973	113	1941	143	1926
24	1955	54	1984	84	1936	114	1938	144	1959
25	1965	55	1933	85	1957	115	1951	145	1969
26	1961	56	1948	86	1964	116	1956	146	1950
27	1945	57	1941	87	1963	117	1965	147	1937
28	1961	58	1941	88	1936	118	1948	148	1985
29	1965	59	1942	89	1955	119	1947	149	1967
30	1933	60	1957	90	1953	120	1983	150	1958

C.3.3 Amostras Totais para o cenário 50x20

Tabela C.16 Amostras finais de dados para o trabalho TGLOBAL no cenário 50x20.

Execução	*Makespan*	Execução	*Makespan*	Execução	*Makespan*	Execução	*Makespan*	Execução	*Makespan*
1	3293	8	3205	15	3294	22	3215	29	3227
2	3254	9	3261	16	3329	23	3247	30	3272
3	3258	10	3234	17	3314	24	3287	31	3241
4	3255	11	3259	18	3261	25	3276	32	3318
5	3301	12	3271	19	3245	26	3233	33	3250
6	3288	13	3252	20	3271	27	3327	34	3250
7	3272	14	3228	21	3276	28	3222	35	3252

Tabela C.17 Amostras finais de dados para o trabalho TRN no cenário 30x15.

Execução	*Makespan*	Execução	*Makespan*	Execução	*Makespan*	Execução	*Makespan*	Execução	*Makespan*
1	3227	19	3240	37	3217	55	3247	73	3251
2	3256	20	3245	38	3237	56	3273	74	3235
3	3282	21	3261	39	3257	57	3256	75	3232
4	3261	22	3241	40	3244	58	3209	76	3254
5	3251	23	3214	41	3195	59	3217	77	3257
6	3234	24	3265	42	3301	60	3257	78	3263
7	3231	25	3209	43	3251	61	3236	79	3274
8	3218	26	3246	44	3272	62	3182	80	3217
9	3226	27	3267	45	3224	63	3249	81	3287
10	3271	28	3253	46	3267	64	3306	82	3240
11	3236	29	3310	47	3272	65	3225	83	3241
12	3214	30	3256	48	3216	66	3286	84	3315
13	3212	31	3294	49	3242	67	3226	85	3263
14	3266	32	3300	50	3229	68	3215	86	3294
15	3309	33	3271	51	3220	69	3260	87	3263
16	3240	34	3256	52	3249	70	3251	88	3250
17	3264	35	3253	53	3274	71	3249	89	3219
18	3238	36	3244	54	3212	72	3203	90	3247

Tabela C.18 Amostras finais de dados para o trabalho TCC no cenário 30x15.

Execução	*Makespan*	Execução	*Makespan*	Execução	*Makespan*	Execução	*Makespan*	Execução	*Makespan*
1	3347	8	3300	15	3360	22	3273	29	3324
2	3253	9	3235	16	3292	23	3294	30	3282
3	3237	10	3275	17	3268	24	3261	31	3269
4	3283	11	3330	18	3321	25	3257	32	3342
5	3246	12	3265	19	3245	26	3224	33	3257
6	3254	13	3288	20	3250	27	3295	34	3330
7	3290	14	3299	21	3331	28	3277	35	3265

Tabela C.19 Amostras finais de dados para o trabalho TBACO no cenário 30x15.

Execução	*Makespan*	Execução	*Makespan*	Execução	*Makespan*	Execução	*Makespan*	Execução	*Makespan*
1	3258	19	3228	37	3217	55	3184	73	3239
2	3232	20	3208	38	3231	56	3232	74	3275
3	3199	21	3240	39	3209	57	3232	75	3186
4	3212	22	3200	40	3227	58	3280	76	3237
5	3196	23	3203	41	3208	59	3222	77	3210
6	3238	24	3218	42	3250	60	3224	78	3239
7	3269	25	3232	43	3225	61	3257	79	3236
8	3250	26	3260	44	3236	62	3248	80	3212
9	3225	27	3261	45	3220	63	3251	81	3266
10	3241	28	3243	46	3207	64	3187	82	3257
11	3279	29	3287	47	3252	65	3213	83	3241
12	3222	30	3266	48	3245	66	3224	84	3225
13	3220	31	3271	49	3208	67	3271	85	3260
14	3198	32	3242	50	3232	68	3224	86	3254
15	3265	33	3226	51	3246	69	3271	87	3262
16	3258	34	3276	52	3258	70	3244	88	3204
17	3256	35	3204	53	3254	71	3243	89	3240
18	3227	36	3281	54	3205	72	3254	90	3228

C.4 Testes de Hipótese usando *t-student*

Para o teste de hipótese usando *t-student* algumas considerações devem ser feitas:

- Definir nível de significância, definido como 0,05 ou 5%;
- Garantir que as amostras existem em número suficiente para assegurar um poder de teste definido em 95%, obtido pelos cálculos de tamanho amostral da seção 2;
- Garantir que as amostras possuem distribuição normal suficientemente boa para que se possa inferir, assumido visto que as populações amostrais são superiores a 30 amostras;
- Garantir que as amostras são independentes, obtido pelo fato de serem geradas aleatoriamente.

As tabelas das próximas seções trazem todos os resultados de cada teste de hipótese, lembrando que o grupo 1 é o TBACO e o grupo 2 é o trabalho sendo comparado.

A hipótese nula é rejeitada sempre que o valor de p for menor que 0,05.

A hipótese a ser testada é se a média dos *makespan* de TBACO é menor que a média do *makespan* do trabalho com quem esta sendo comparado. Porém o suplemento usado para o cálculo do teste de hipóteses só permite as hipóteses nulas na forma: igual, maior ou igual e menor ou igual.

Dois caminhos são possíveis para se produzir o efeito de aceitar menor que:

- Aceitar que a média do TBACO é menor ou igual ao concorrente e rejeitar que a média do TBACO é igual ao do trabalho concorrente;
- Rejeitar que a média do trabalho comparado é menor ou igual à média de TBACO.

A segunda opção soa estranha, pois corrobora a tese pelo falseamento de uma hipótese que é o contrário da hipótese de TBACO.

Portanto escolheu-se a primeira opção.

Para cada comparação dois testes forma executados. A primeira hipótese nula é a de que o *makespan* médio obtido pelo TBACO é menor ou igual ao do trabalho em comparação. Sua não rejeição corrobora TBACO como menor ou igual ao concorrente. A segunda hipótese

nula é a de que o *makespan* médio do TBACO e do concorrente são iguais. A sua rejeição implica em que são diferentes.

Pela aceitação da primeira hipótese nula e rejeição da segunda hipótese nula, corrobora-se que a média dos *makespans* de TBACO é menor do que a média dos *makespans* do trabalho concorrente. A prova deste parágrafo encontra-se no Apêndice B.

Nas próximas subseções são apresentadas as tabelas com os dados dos testes de hipóteses feitos, por cenário. Algumas conclusões são indicadas a cada apresentação.

C.4.1 Testes de hipótese para o cenário 15x15

Analisando-se as Tabelas C.20 e C.21 nota-se que quando a hipótese nula é que o *makespan* médio de TBACO é igual ao *makespan* médio do TGLOBAL, o p-valor é bem menor que 0,05, caracterizando a não aceitação desta hipótese nula; e que quando a hipótese nula é que o *makespan* médio de TBACO é menor ou igual ao *makespan* médio do TGLOBAL, o p-valor aproxima-se de um, caracterizando a aceitação desta hipótese nula. Portanto para esta comparação conclui-se que o *makespan* médio de TBACO é menor que o *makespan* médio do TGLOBAL.

A mesma análise pode ser feita para as Tabelas C.22 e C.23 e para as Tabelas C.24 e C.25. Portanto para estas comparações conclui-se que o *makespan* médio de TBACO é menor que o *makespan* médio de TRN e de TCC.

Tabela C.20 Valores do teste de hipótese "*makespan* médio de TBACO igual ao *makespan* médio de TGLOBAL", para o cenário 15x15.

Informação	Valor
T	-4,422225612
Graus de Liberdade	62,24244193
P-valor	4,00E-05
Média no grupo 1:	1318,2
Média no grupo 2:	1332,257143
Desvio padrão amostral do grupo 1:	11,09265788
Desvio padrão amostral do grupo 2:	15,18579611
Hipótese Alternativa: Diferente de	0
Intervalo de Confiança	95%
Limite Inferior	-20,41087819
Limite Superior	-7,703407524

Tabela C.21 Valores do teste de hipótese "*makespan* médio de TBACO menor ou igual ao *makespan* médio de TGLOBAL", para o cenário 15x15.

Informação	Valor
T	-4,422225612
Graus de Liberdade	62,24244193
P-valor	0,999980018
Média no grupo 1:	1318,2
Média no grupo 2:	1332,257143
Desvio padrão amostral do grupo 1:	11,09265788
Desvio padrão amostral do grupo 2:	15,18579611
Hipótese Alternativa: Maior que	0
Intervalo de Confiança	95%
Limite Inferior	-19,36471668

Tabela C.22 Valores do teste de hipótese "*makespan* médio de TBACO igual ao *makespan* médio de TRN", para o cenário 15x15.

Informação	Valor
T	-4,210586148
Graus de Liberdade	137,016857
P-valor	4,59E-05
Média no grupo 1:	1316,428571
Média no grupo 2:	1324,842857
Desvio padrão amostral do grupo 1:	12,31303237
Desvio padrão amostral do grupo 2:	11,31067947
Hipótese Alternativa: Diferente de	0
Intervalo de Confiança	95%
Limite Inferior	-12,36590964
Limite Superior	-4,46266179

Tabela C.23 Valores do teste de hipótese "*makespan* médio de TBACO menor ou igual ao *makespan* médio de TRN", para o cenário 15x15.

Informação	Valor
T	-4,210586148
Graus de Liberdade	137,016857
P-valor	0,999977072
Média no grupo 1:	1316,428571
Média no grupo 2:	1324,842857
Desvio padrão amostral do grupo 1:	12,31303237
Desvio padrão amostral do grupo 2:	11,31067947
Hipótese Alternativa: Maior que	0
Intervalo de Confiança	95%
Limite Inferior	-11,72367856

Tabela C.24 Valores do teste de hipótese "*makespan* médio de TBACO igual ao *makespan* médio de TCC", para o cenário 15x15.

Informação	Valor
T	-3,567723612
Graus de Liberdade	113,4237821
P-valor	0,000528832
Média no grupo 1:	1316,483333
Média no grupo 2:	1323,716667
Desvio padrão amostral do grupo 1:	12,16899598
Desvio padrão amostral do grupo 2:	9,927007617
Hipótese Alternativa: Diferente de	0
Intervalo de Confiança	95%
Limite Inferior	-11,24988844
Limite Superior	-3,216778225

Tabela C.25 Valores do teste de hipótese "*makespan* médio de TBACO menor ou igual ao *makespan* médio de TCC", para o cenário 15x15.

Informação	Valor
T	-3,567723612
Graus de Liberdade	113,4237821
P-valor	0,999735584
Média no grupo 1:	1316,483333
Média no grupo 2:	1323,716667
Desvio padrão amostral do grupo 1:	12,16899598
Desvio padrão amostral do grupo 2:	9,927007617
Hipótese Alternativa: Maior que	0
Intervalo de Confiança	95%
Limite Inferior	-10,59563211

C.4.2 Testes de hipótese para o cenário 30x15

Analisando-se as Tabelas C.26 e C.27 nota-se que quando a hipótese nula é que o *makespan* médio de TBACO é igual ao *makespan* médio do TGLOBAL, o p-valor é bem menor que 0,05, caracterizando a não aceitação desta hipótese nula; e que quando a hipótese nula é que o *makespan* médio de TBACO é menor ou igual ao *makespan* médio do TGLOBAL, o p-valor aproxima-se de um, caracterizando a aceitação desta hipótese nula. Portanto para esta comparação conclui-se que o *makespan* médio de TBACO é menor que o *makespan* médio do TGLOBAL.

A mesma análise pode ser feita para as Tabelas C.28 e C.29 e para as Tabelas C.30 e C.31. Portanto para estas comparações conclui-se que o *makespan* médio de TBACO é menor que o *makespan* médio de TRN e de TCC.

Tabela C.26 Valores do teste de hipótese "*makespan* médio de TBACO igual ao *makespan* médio de TGLOBAL", para o cenário 30x15.

Informação	Valor
T	-6,744999089
Graus de Liberdade	54,62797755
P-valor	1,02E-08
Média no grupo 1:	1956,857143
Média no grupo 2:	1985,285714
Desvio padrão amostral do grupo 1:	12,53264644
Desvio padrão amostral do grupo 2:	21,55645315
Hipótese Alternativa: Diferente de	0
Intervalo de Confiança	95%
Limite Inferior	-36,87643801
Limite Superior	-19,98070485

Tabela C.27 Valores do teste de hipótese "*makespan* médio de TBACO menor ou igual ao *makespan* médio de TGLOBAL", para o cenário 30x15.

Informação	Valor
T	-6,744999089
Graus de Liberdade	54,62797755
P-valor	0,999999995
Média no grupo 1:	1956,857143
Média no grupo 2:	1985,285714
Desvio padrão amostral do grupo 1:	12,53264644
Desvio padrão amostral do grupo 2:	21,55645315
Hipótese Alternativa: Maior que	0
Intervalo de Confiança	95%
Limite Inferior	-35,48083557

Tabela C.28 Valores do teste de hipótese "*makespan* médio de TBACO igual ao *makespan* médio de TRN", para o cenário 30x15.

Informação	Valor
T	-5,722044406
Graus de Liberdade	163,5923056
P-valor	4,90E-08
Média no grupo 1:	1955,355556
Média no grupo 2:	1970,1
Desvio padrão amostral do grupo 1:	14,49549416
Desvio padrão amostral do grupo 2:	19,68404932
Hipótese Alternativa: Diferente de	0
Intervalo de Confiança	95%
Limite Inferior	-19,83247813
Limite Superior	-9,656410763

Tabela C.29 Valores do teste de hipótese "*makespan* médio de TBACO menor ou igual ao *makespan* médio de TRN", para o cenário 30x15.

Informação	Valor
T	-5,722044406
Graus de Liberdade	163,5923056
P-valor	0,999999976
Média no grupo 1:	1955,355556
Média no grupo 2:	1970,1
Desvio padrão amostral do grupo 1:	14,49549416
Desvio padrão amostral do grupo 2:	19,68404932
Hipótese Alternativa: Maior que	0
Intervalo de Confiança	95%
Limite Inferior	-19,00700745

Tabela C.30 Valores do teste de hipótese "*makespan* médio de TBACO igual ao *makespan* médio de TCC", para o cenário 30x15.

Informação	Valor
T	-4,482093963
Graus de Liberdade	279,7001237
P-valor	1,08E-05
Média no grupo 1:	1954,52
Média no grupo 2:	1963,82
Desvio padrão amostral do grupo 1:	15,50177301
Desvio padrão amostral do grupo 2:	20,13680393
Hipótese Alternativa: Diferente de	0
Intervalo de Confiança	95%
Limite Inferior	-13,38444794
Limite Superior	-5,215552062

Tabela C.31 Valores do teste de hipótese "*makespan* médio de TBACO menor ou igual ao *makespan* médio de TCC", para o cenário 30x15.

Informação	Valor
T	-4,482093963
Graus de Liberdade	279,7001237
P-valor	0,999994605
Média no grupo 1:	1954,52
Média no grupo 2:	1963,82
Desvio padrão amostral do grupo 1:	15,50177301
Desvio padrão amostral do grupo 2:	20,13680393
Hipótese Alternativa: Maior que	0
Intervalo de Confiança	95%
Limite Inferior	-12,72428634

C.4.3 Testes de hipótese para o cenário 50x20

Analisando-se as Tabelas C.32 e C.33 nota-se que quando a hipótese nula é que o *makespan* médio de TBACO é igual ao *makespan* médio do TGLOBAL, o p-valor é bem menor que 0,05, caracterizando a não aceitação desta hipótese nula; e que quando a hipótese nula é que o *makespan* médio de TBACO é menor ou igual ao *makespan* médio do TGLOBAL, o p-valor aproxima-se de um, caracterizando a aceitação desta hipótese nula. Portanto para esta comparação conclui-se que o *makespan* médio de TBACO é menor que o *makespan* médio do TGLOBAL.

A mesma análise pode ser feita para as Tabelas C.34 e C.35 e para as Tabelas C.36 e C.37. Portanto para estas comparações conclui-se que o *makespan* médio de TBACO é menor que o *makespan* médio de TRN e de TCC.

Tabela C.32 Valores do teste de hipótese "*makespan* médio de TBACO igual ao *makespan* médio de TGLOBAL", para o cenário 50x20.

Informação	Valor
T	-3,868043559
Graus de Liberdade	66,15208027
P-valor	0,000253185
Média no grupo 1:	3237,428571
Média no grupo 2:	3263,942857
Desvio padrão amostral do grupo 1:	26,16945426
Desvio padrão amostral do grupo 2:	30,97906585
Hipótese Alternativa: Diferente de	0
Intervalo de Confiança	95%
Limite Inferior	-40,19955248
Limite Superior	-12,82901895

Tabela C.33 Valores do teste de hipótese "*makespan* médio de TBACO menor ou igual ao *makespan* médio de TGLOBAL", para o cenário 50x20.

Informação	Valor
T	-3,868043559
Graus de Liberdade	66,15208027
P-valor	0,999873407
Média no grupo 1:	3237,428571
Média no grupo 2:	3263,942857
Desvio padrão amostral do grupo 1:	26,16945426
Desvio padrão amostral do grupo 2:	30,97906585
Hipótese Alternativa: Maior que	0
Intervalo de Confiança	95%
Limite Inferior	-37,9494086

Tabela C.34 Valores do teste de hipótese "*makespan* médio de TBACO igual ao *makespan* médio de TRN", para o cenário 50x20.

Informação	Valor
T	-3,270863337
Graus de Liberdade	175,9255195
P-valor	0,001290177
Média no grupo 1:	3236,144444
Média no grupo 2:	3248,766667
Desvio padrão amostral do grupo 1:	24,44096814
Desvio padrão amostral do grupo 2:	27,25619196
Hipótese Alternativa: Diferente de	0
Intervalo de Confiança	95%
Limite Inferior	-20,23808998
Limite Superior	-5,006354469

Tabela C.35 Valores do teste de hipótese "*makespan* médio de TBACO menor ou igual ao *makespan* médio de TRN", para o cenário 50x20.

Informação	Valor
T	-3,270863337
Graus de Liberdade	175,9255195
P-valor	0,999354911
Média no grupo 1:	3236,144444
Média no grupo 2:	3248,766667
Desvio padrão amostral do grupo 1:	24,44096814
Desvio padrão amostral do grupo 2:	27,25619196
Hipótese Alternativa: Maior que	0
Intervalo de Confiança	95%
Limite Inferior	-19,00329475

Tabela C.36 Valores do teste de hipótese "*makespan* médio de TBACO igual ao *makespan* médio de TCC", para o cenário 50x20.

Informação	Valor
T	-6,256553325
Graus de Liberdade	63,21407075
P-valor	3,83E-08
Média no grupo 1:	3237,428571
Média no grupo 2:	3283,4
Desvio padrão amostral do grupo 1:	26,16945426
Desvio padrão amostral do grupo 2:	34,7098901
Hipótese Alternativa: Diferente de	0
Intervalo de Confiança	95%
Limite Inferior	-60,65371077
Limite Superior	-31,28914637

Tabela C.37 Valores do teste de hipótese "*makespan* médio de TBACO menor ou igual ao *makespan* médio de TCC", para o cenário 50x20.

Informação	Valor
T	-6,256553325
Graus de Liberdade	63,21407075
P-valor	0,999999981
Média no grupo 1:	3237,428571
Média no grupo 2:	3283,4
Desvio padrão amostral do grupo 1:	26,16945426
Desvio padrão amostral do grupo 2:	34,7098901
Hipótese Alternativa: Maior que	0
Intervalo de Confiança	95%
Limite Inferior	-58,23711585

Apêndice D

GRÁFICOS E TABELAS ADICIONAIS

Neste apêndice, são apresentados alguns gráficos e tabelas produzidos a partir dos dados gerados pela execução dos experimentos. Podem ser vistos como um suporte para algumas das propostas de trabalho futuro.

D.1 Gráficos de comportamento do Algoritmo Genético

Nesta seção são apresentados os gráficos de comportamento do Algoritmo Genético para cada trabalho em cada cenário. Os gráficos foram reunidos em subseções, em cada uma apenas os gráficos referentes a um trabalho especifico.

As curvas apresentadas indicam o comportamento da variação de:

- pior makespan da geração;
- makespan médio da geração; e
- melhor makespan da geração.

Estes makespans, por geração, são resultados das médias de todas as execuções do trabalho.

Percebe-se que em todos, as curvas dos piores makespan e makespan médio possuem uma tendência de queda, mas com flutuações. A curva do menor makespan converge continuamente.

D.1.1 Comportamento do AG para o trabalho TGLOBAL

Nesta subseção estão os gráficos do comportamento do AG para o trabalho TGLOBAL nos cenários da avaliação.

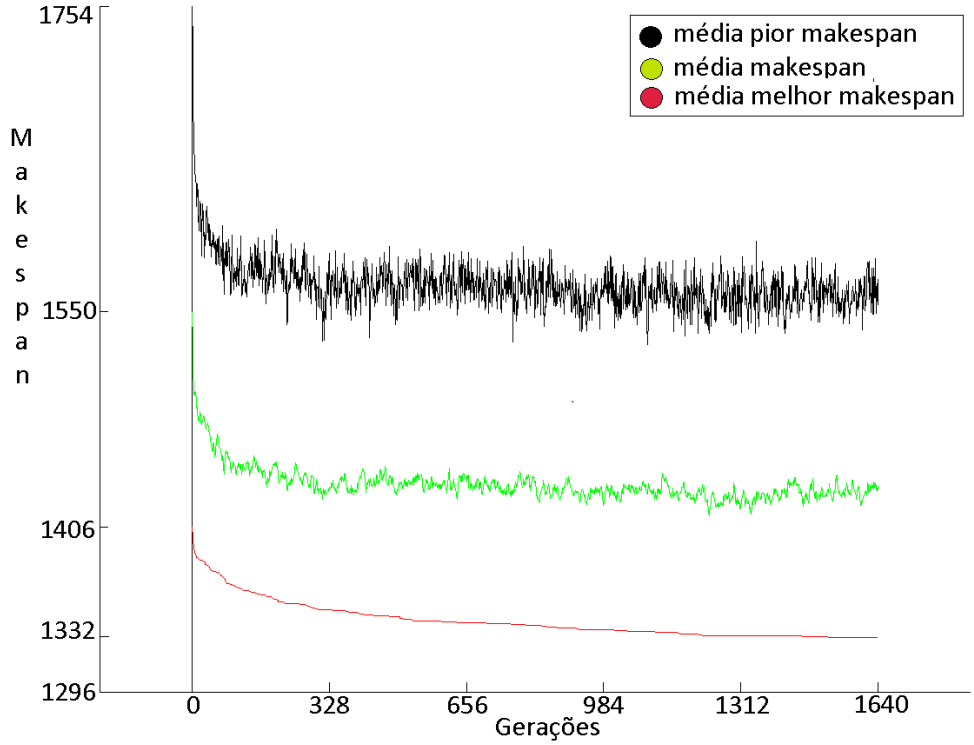

Figura D.1 Gráfico do comportamento da convergência do pior makespan, makespan médio e melhor makespan ao longo das gerações para o trabalho TGLOBAL no cenário 15 x 15.

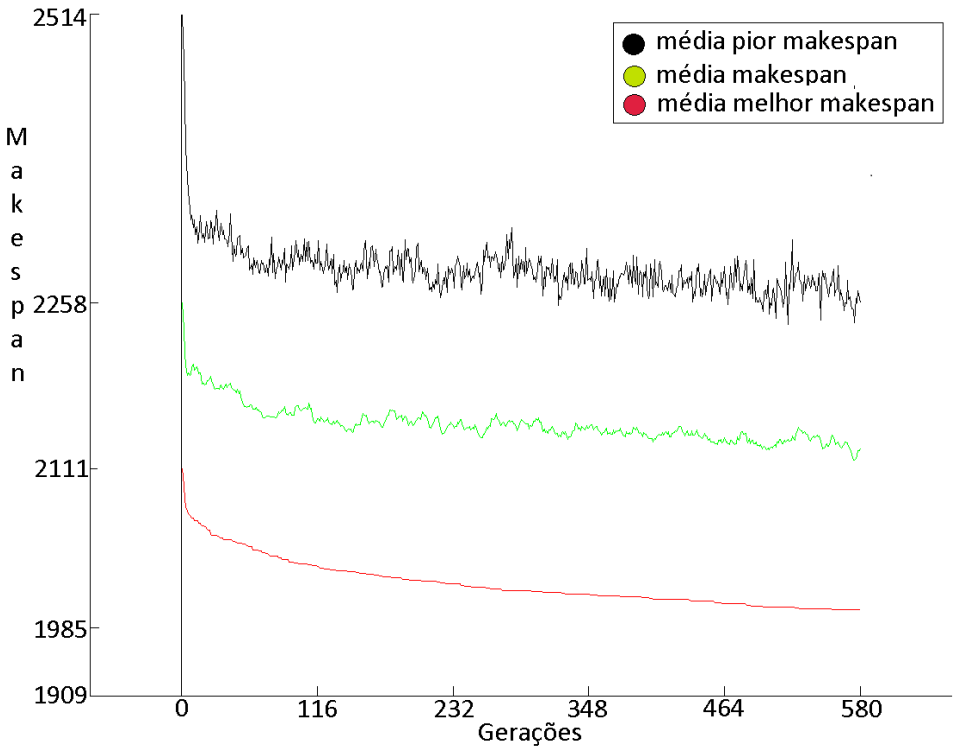

Figura D.2 Gráfico do comportamento da convergência do pior makespan, makespan médio e melhor makespan ao longo das gerações para o trabalho TGLOBAL no cenário 30 x 15.

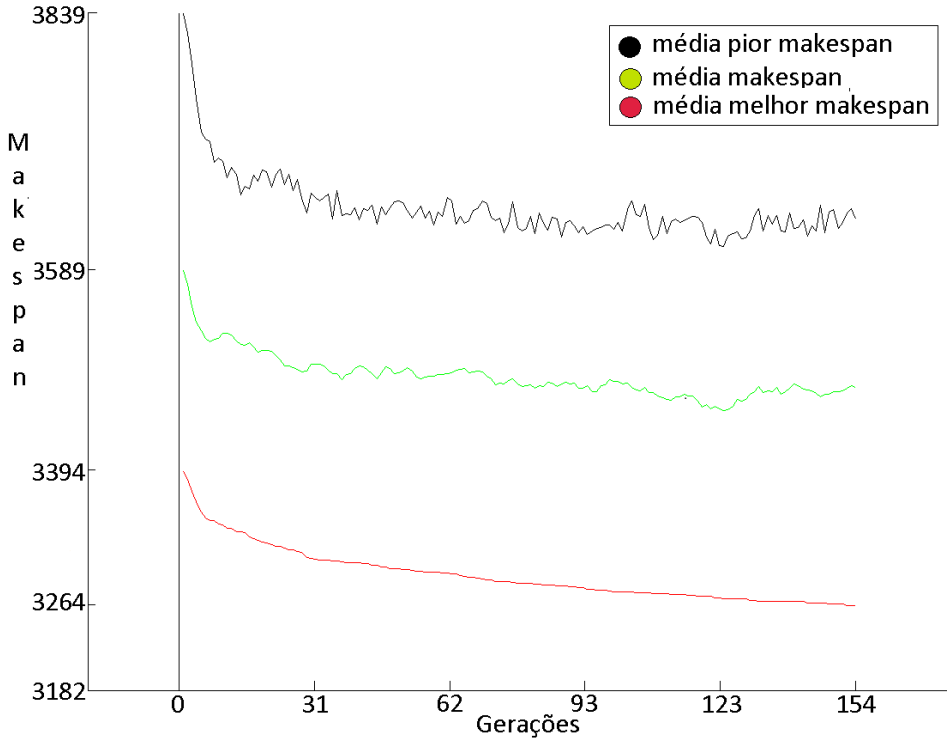

Figura D.3 Gráfico do comportamento da convergência do pior makespan, makespan médio e melhor makespan ao longo das gerações para o trabalho TGLOBAL no cenário 50 x 20.

D.1.2 Comportamento do AG para o trabalho TRN

Nesta subseção estão os gráficos do comportamento do AG para o trabalho TRN nos cenários da avaliação.

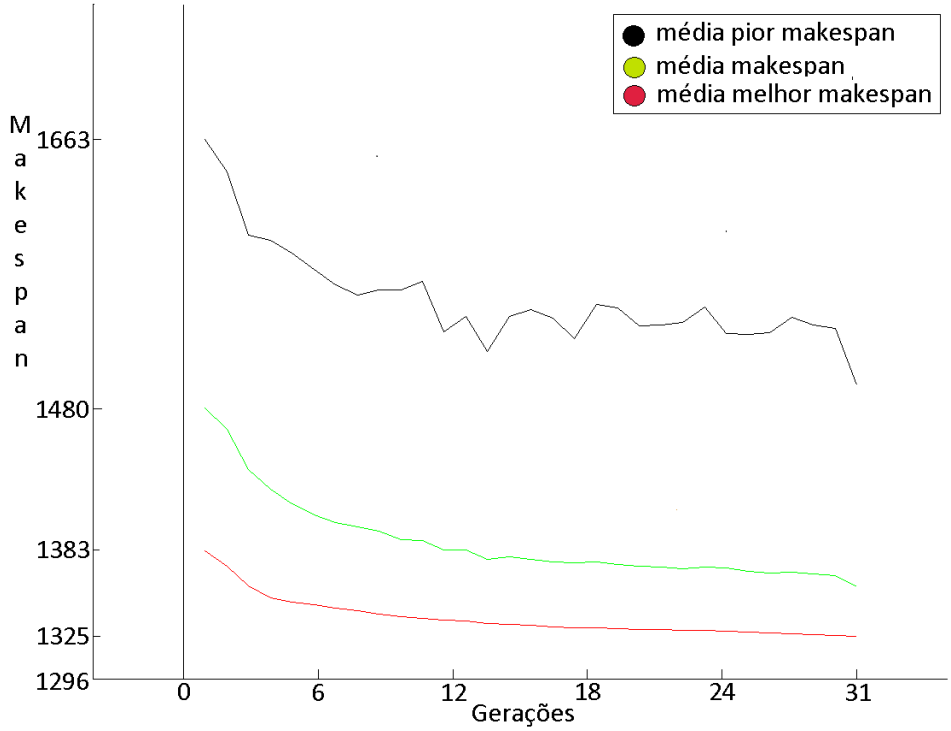

Figura D.4 Gráfico do comportamento da convergência do pior makespan, makespan médio e melhor makespan ao longo das gerações para o trabalho TRN no cenário 15 x 15.

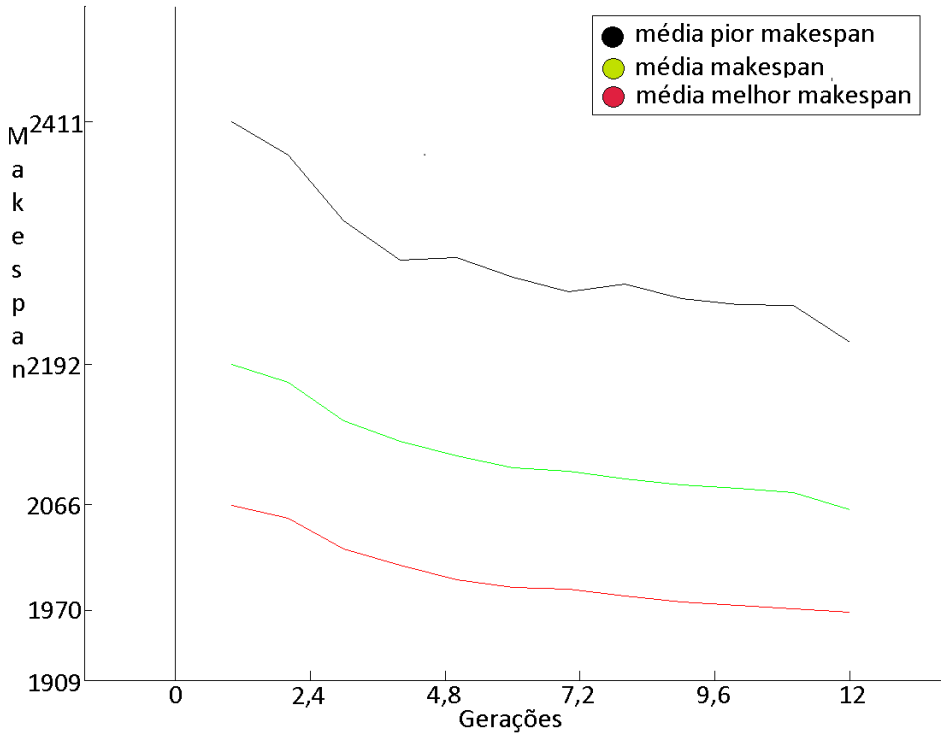

Figura D.5 Gráfico do comportamento da convergência do pior makespan, makespan médio e melhor makespan ao longo das gerações para o trabalho TRN no cenário 30 x 15.

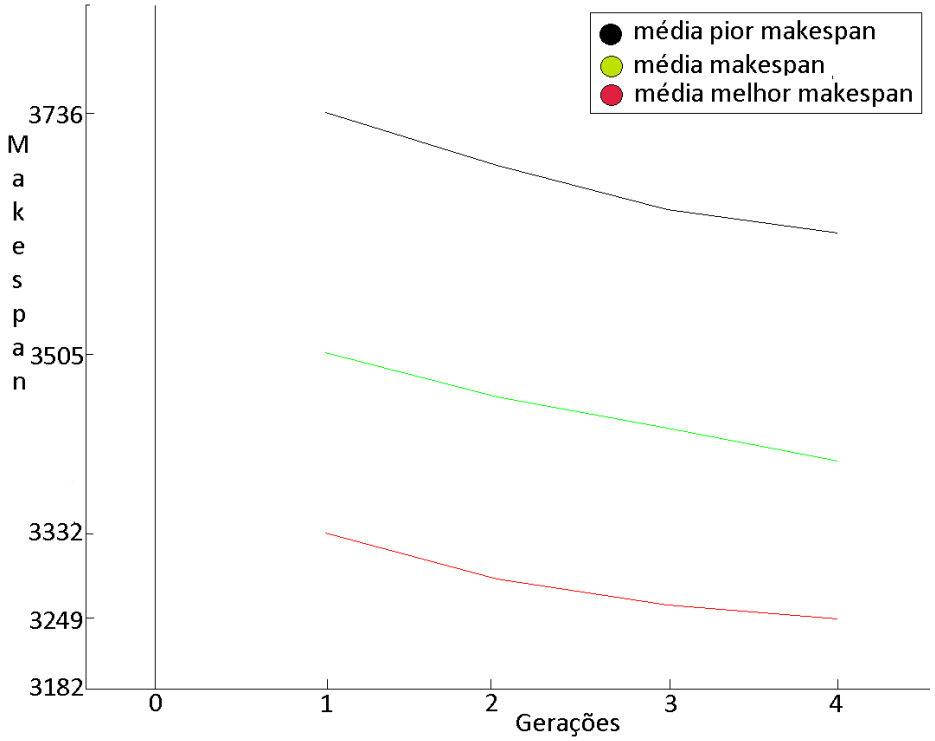

Figura D.6 Gráfico do comportamento da convergência do pior makespan, makespan médio e melhor makespan ao longo das gerações para o trabalho TRN no cenário 50 x 20.

D.1.3 Comportamento do AG para o trabalho TCC

Nesta subseção estão os gráficos do comportamento do AG para o trabalho TCC nos cenários da avaliação.

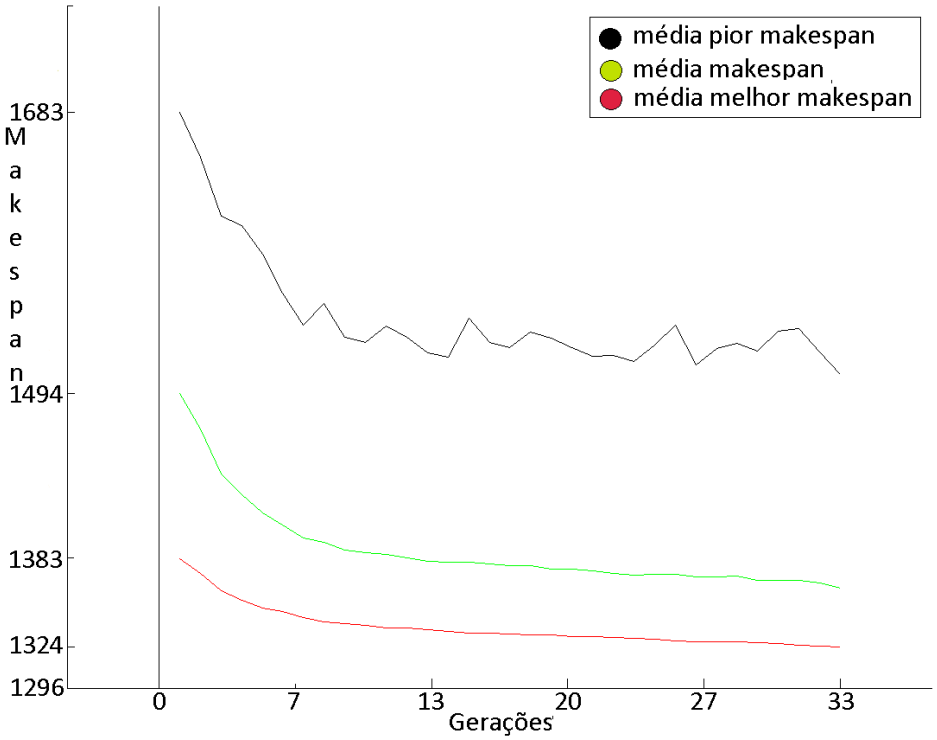

Figura D.7 Gráfico do comportamento da convergência do pior makespan, makespan médio e melhor makespan ao longo das gerações para o trabalho TCC no cenário 15 x 15.

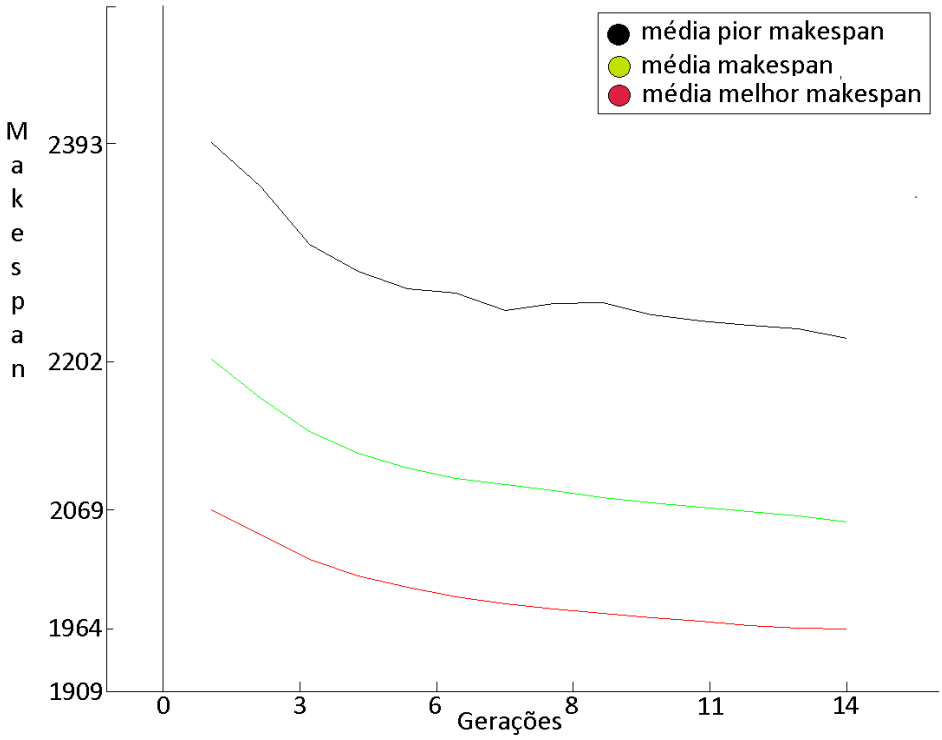

Figura D.8 Gráfico do comportamento da convergência do pior makespan, makespan médio e melhor makespan ao longo das gerações para o trabalho TCC no cenário 30 x 15.

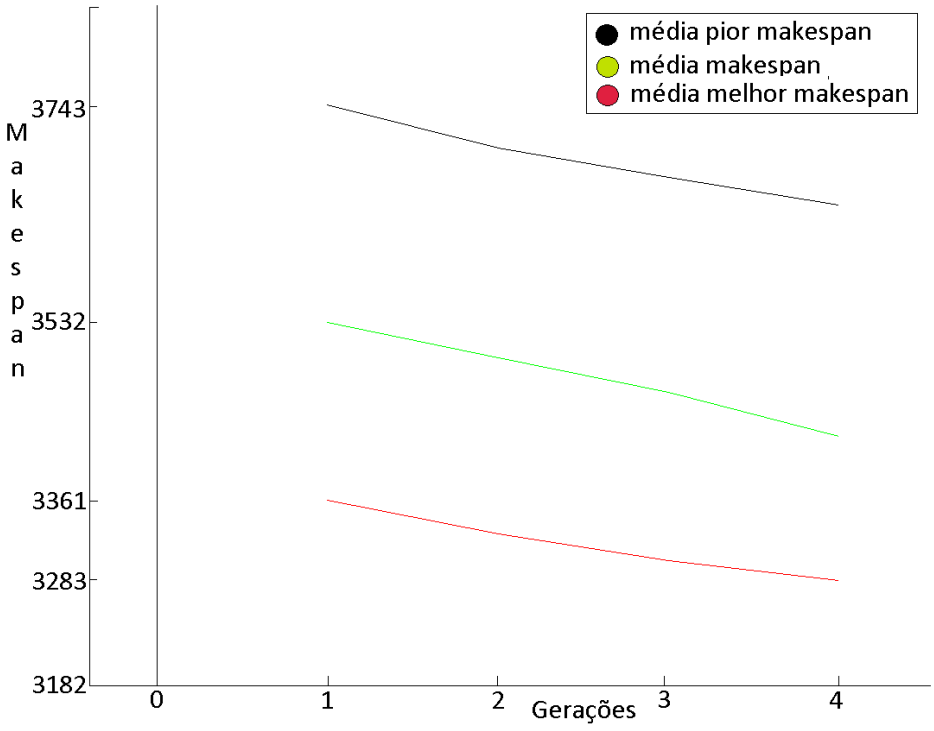

Figura D.9 Gráfico do comportamento da convergência do pior makespan, makespan médio e melhor makespan ao longo das gerações para o trabalho TCC no cenário 50 x 20.

D.1.4 Comportamento do AG para o trabalho TBACO

Nesta subseção estão os gráficos do comportamento do AG para o trabalho TRN nos cenários da avaliação.

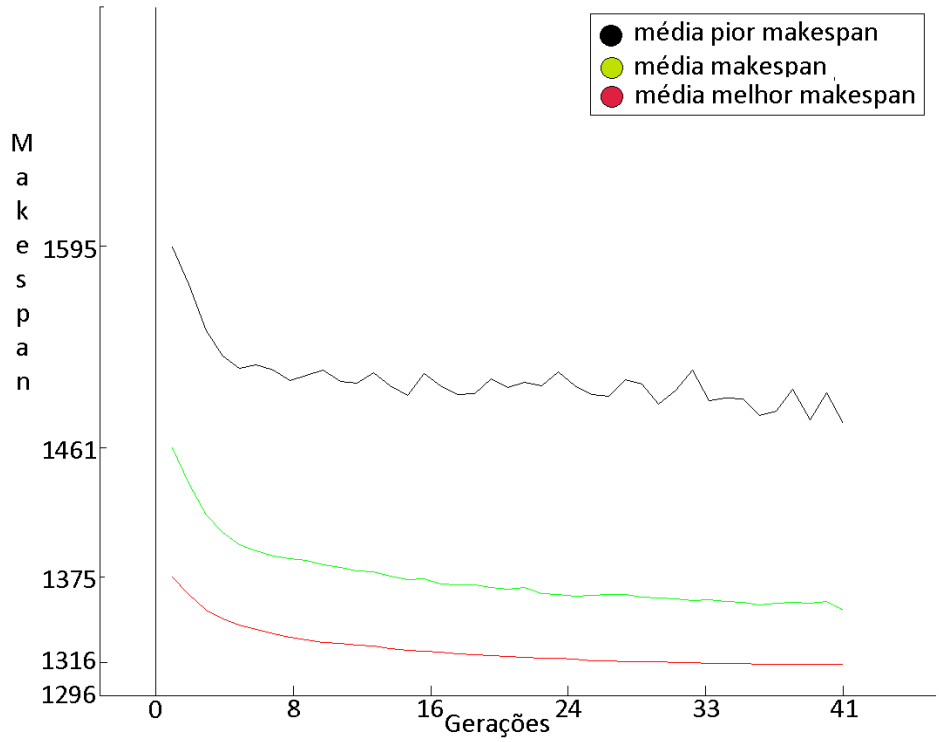

Figura D.10 Gráfico do comportamento da convergência do pior makespan, makespan médio e melhor makespan ao longo das gerações para o trabalho TBACO no cenário 15 x 15.

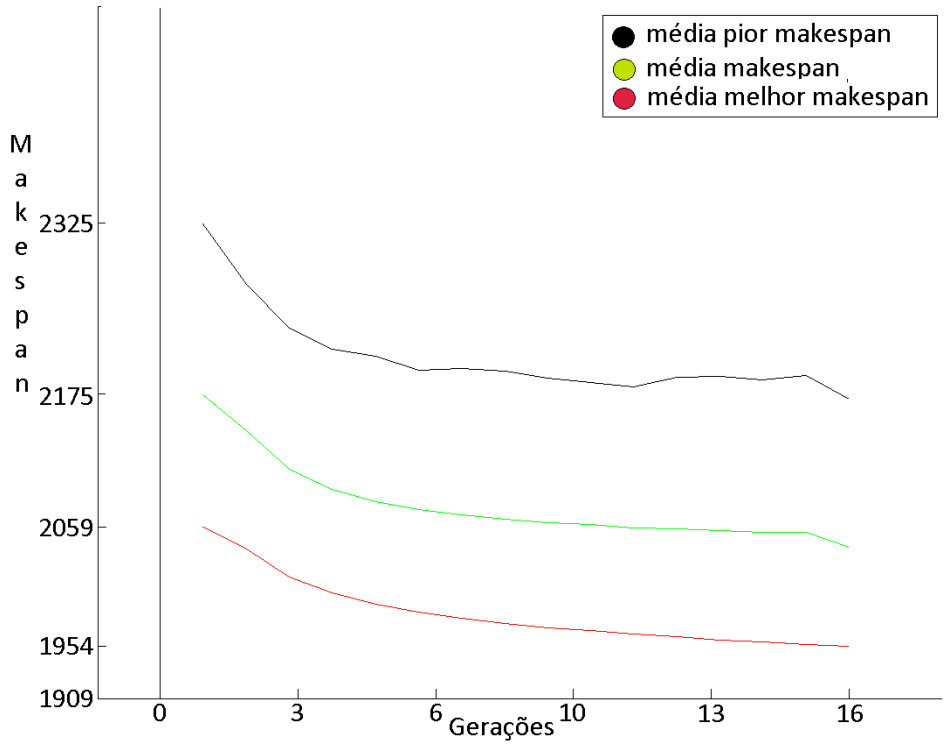

Figura D.11 Gráfico do comportamento da convergência do pior makespan, makespan médio e melhor makespan ao longo das gerações para o trabalho TBACO no cenário 30 x 15.

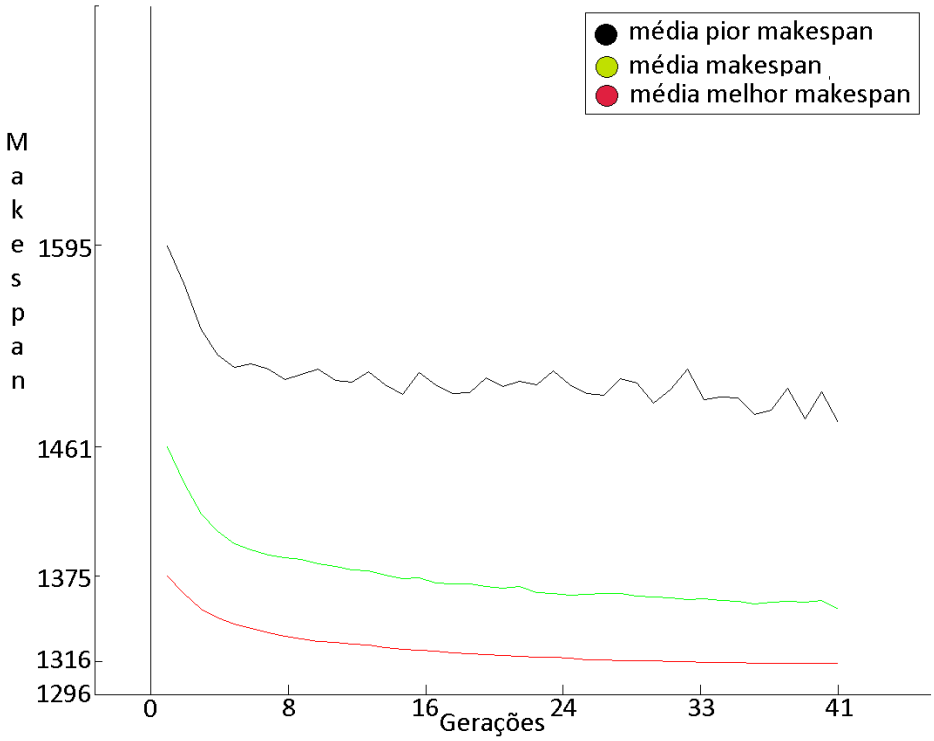

Figura D.12 Gráfico do comportamento da convergência do pior makespan, makespan médio e melhor makespan ao longo das gerações para o trabalho TBACO no cenário 50 x 20.

D.2 Gráficos das convergências por cenário

Nesta seção são apresentados os gráficos de comparação da convergência para cada cenário. As curvas usam o melhor *makespan* médio por trabalho, a cada fração de tempo (10 segundos), compondo um conjunto de trinta pontos por trabalho.

Nota-se que todos os trabalhos convergem, e que o TBACO tem melhores resultados nos 3 cenários.

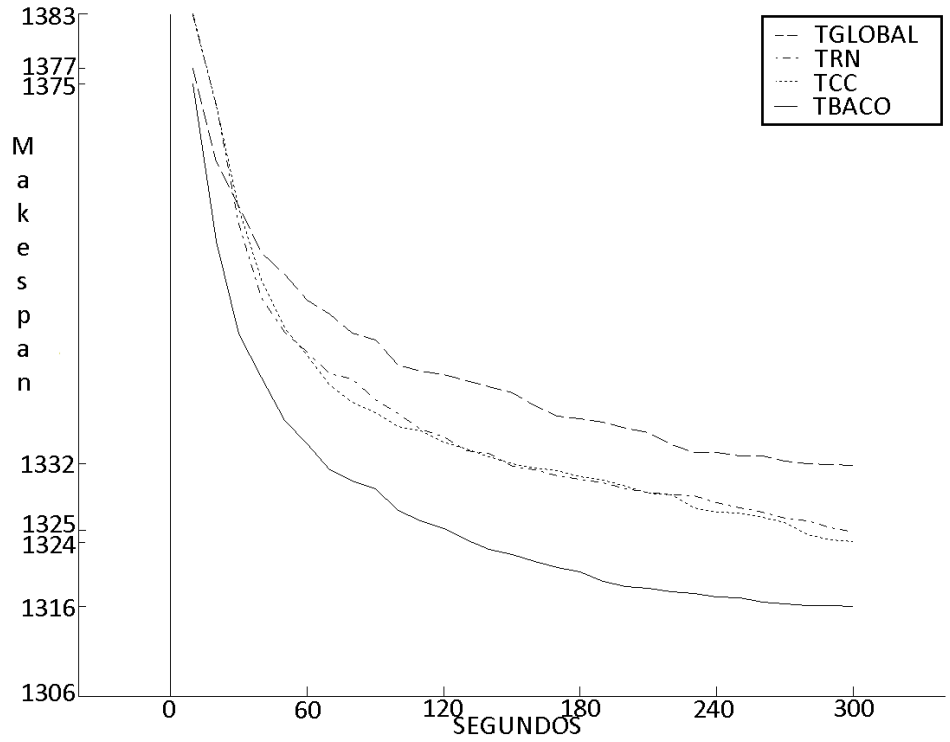

Figura D.13 Gráfico do comportamento das convergências dos melhores makespans de cada trabalho, ao longo do tempo, no cenário 15 x 15.

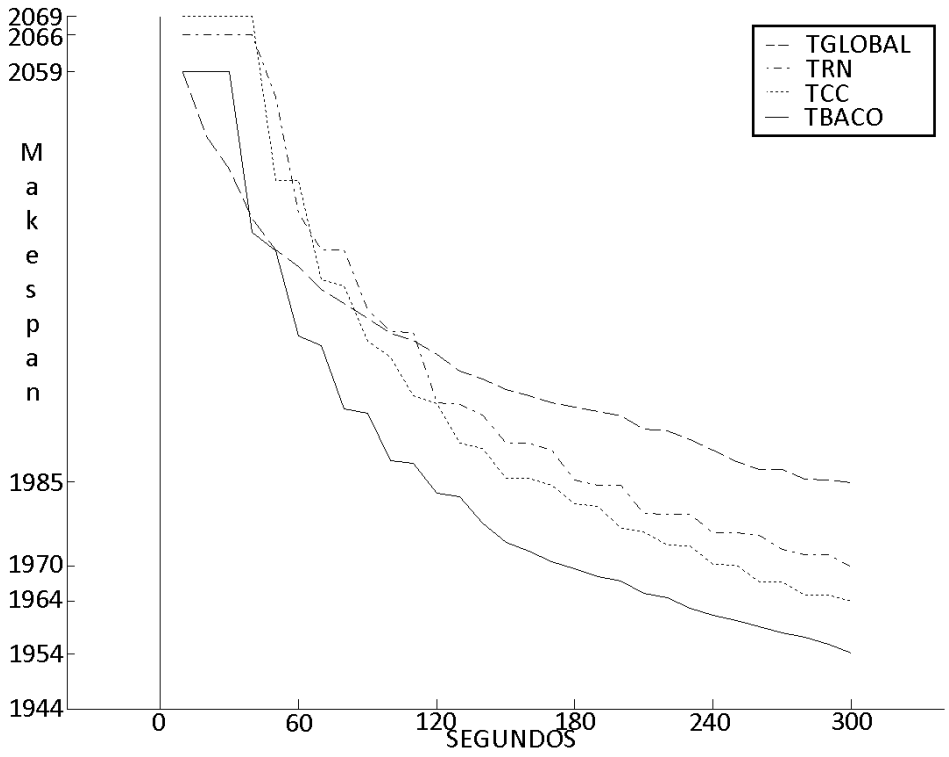

Figura D.14 Gráfico do comportamento das convergências dos melhores makespans de cada trabalho, ao longo do tempo, no cenário 30 x 15.

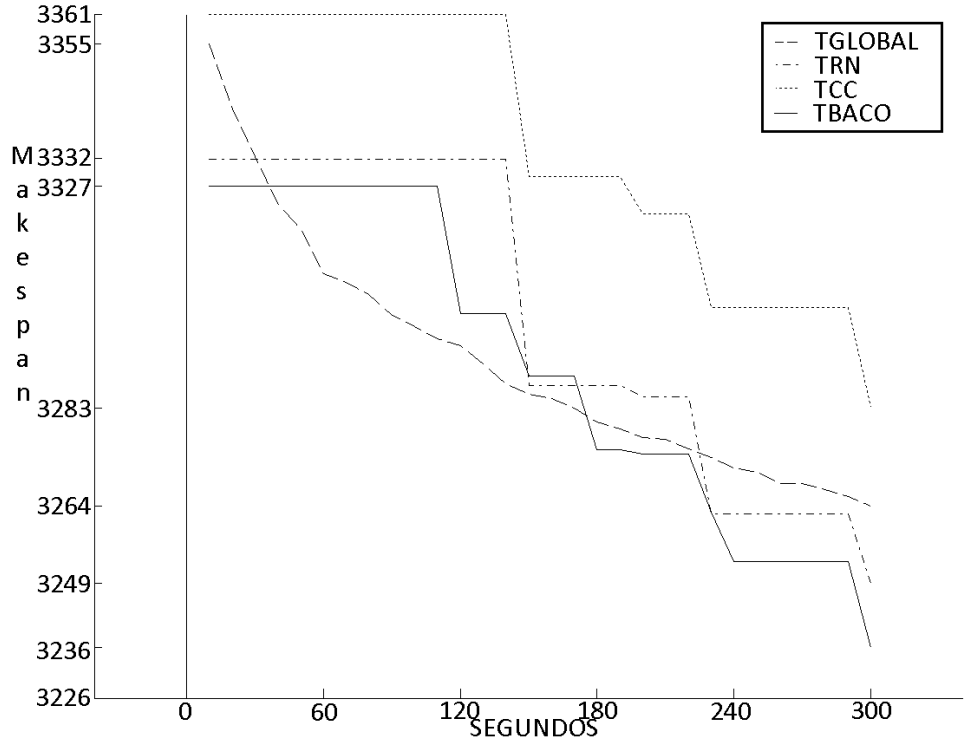

Figura D.15 Gráfico do comportamento das convergências dos melhores makespans de cada trabalho, ao longo do tempo, no cenário 50 x 20.

D.3 Tabelas de desempenho dos operadores de vizinhança

Tabela D.1 Desempenho dos operadores de vizinhança nas soluções dadas como entrada para os trabalhos e cenários desta tese.

Trabalho/ Cenário	Entrados na BL	Melhorados	Melhorados %	Insert	Insert %	Swap	Swap %	2-opt	2-opt %
TRN 15/15	64.139	34.330	53,52	6.985	20,35	9.644	28,09	17.701	51,56
TRN 30 x 15	29.111	24.029	82,54	5.886	24,50	7.435	30,94	10.708	44,56
TRN 50 x 20	9.484	8.773	92,50	1.571	17,91	2.136	24,35	5.066	57,75
TCC 15 x 15	58.225	32.115	55,16	13.612	42,39	8.194	25,51	10.309	32,10
TCC 30 x 15	58.669	50.157	85,49	17.978	35,84	14.189	28,29	17.990	35,87
TCC 50 x 20	3.700	3.455	93,38	998	28,89	943	27,29	1.514	43,82
TBACO 15 x 15	84.392	49.479	58,63	14.972	30,26	16.091	32,52	18.416	37,22
TBACO 30 x 15	68.360	55.923	81,81	15.897	28,43	17.644	31,55	22.382	40,02
TBACO 50 x 20	10.750	10.165	94,56	1.758	17,29	2.136	21,01	6.271	61,69
Médias			77,51		27,32		27,73		44,95

A Tabela D-1 mostra os desempenhos dos operadores de vizinhança nas soluções dadas, tanto no geral como especificando que operador obteve o melhor resultado por solução

Nesta tabela percebe-se:

- A busca local efetivamente tem um alto desempenho, uma média de 77 % das soluções submetidas melhoradas;
- O desempenho do TBACO foi levemente superior aos demais trabalhos (coluna 4), com maior desempenho nos cenários 15 x 15 e 50 x 20;
- Entre os operadores, o operador 2-opt teve um desempenho melhor que os demais operadores, com uma média de 44% contra 27%.

Tabela D.2 Desempenho dos operadores de vizinhança em todas as operações nos trabalhos e cenários deste tese.

Trabalho/ Cenário	Operações efetuadas	Melhoradas %	Insert	%	Swap	%	2-opt	%
TRN 15/15	1.924.170	7,38	25.970	18,28	43.093	30,33	73.021	51,39
TRN 30 x 15	873.330	17,74	38.726	25,00	53.807	34,73	62.391	40,27
TRN 50 x 20	284.520	29,25	21.717	26,10	27.838	33,45	33.656	40,45
TCC 15 x 15	1.746.750	10,98	65.803	34,30	54.128	28,22	71.895	37,48
TCC 30 x 15	1.760.070	23,55	144.160	34,79	126.664	30,56	143.589	34,65
TCC 50 x 20	111.000	34,86	13.100	33,86	11.960	30,91	13.632	35,23
TBACO 15 x 15	2.531.760	8,76	58.012	26,17	76.651	34,58	87.016	39,25
TBACO 30 x 15	2.050.800	15,69	93.336	29,01	116.915	36,34	111.469	34,65
TBACO 50 x 20	322.500	29,85	26.324	27,34	31.846	33,08	38.099	39,58
Médias		19,78		28,32		32,47		39,21

A Tabela D-2 mostra os desempenhos dos operadores de vizinhança em todas as operações executadas mostrando os desempenhos percentuais no total (coluna 3), e por operador mostrando seus percentuais considerando somente as melhorias (colunas 5, 7 e 9).

Nesta tabela percebe-se:

- A busca local aumenta sua efetividade na medida em que a dimensão do problema aumenta;
- O operador *2-opt* tem um desempenho 20% superior ao *swap* e 30% superior ao *insert*.