

UNIVERSIDADE FEDERAL DE PERNAMBUCO
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA DE PRODUÇÃO

**OTIMIZAÇÃO MULTI OBJETIVO DA CONFIABILIDADE
VIA SISTEMAS MULTIAGENTES BASEADO EM COLÔNIA
DE FORMIGAS**

DISSERTAÇÃO SUBMETIDA À UFPE
PARA OBTENÇÃO DE GRAU DE MESTRE
POR

ROSANA CAVALCANTE DE OLIVEIRA
Orientador: Prof. Enrique López Droguett, PhD

RECIFE, ABRIL / 2008

Livros Grátis

<http://www.livrosgratis.com.br>

Milhares de livros grátis para download.



UNIVERSIDADE FEDERAL DE PERNAMBUCO
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA DE PRODUÇÃO

PARECER DA COMISSÃO EXAMINADORA
DE DEFESA DE DISSERTAÇÃO DE
MESTRADO ACADÊMICO DE

ROSANA CAVALCANTE DE OLIVEIRA

"Otimização Multiobjetivo da Confiabilidade e Custo para Alocação de Redundâncias e Testes Periódicos Via Colônia de Formigas".


ÁREA DE CONCENTRAÇÃO: PESQUISA OPERACIONAL

A comissão examinadora, composta pelos professores abaixo, sob a presidência do(a) primeiro(a), considera a candidata **ROSANA CAVALCANTE DE OLIVEIRA APROVADA.**


Recife, 24 de abril de 2008.



Prof. ENRIQUE ANDRÉS LÓPEZ DROGUETT, PhD (UFPE)



Prof. LUCIANO NADLER LINS, Doutor (UFPE)



Prof. ENRICO ANTONIO COLOSIMO, PhD (UFMG)

A Deus, sobre todas as coisas.

A minha avó querida, eu sei que estás no céu olhando por mim, e por todos os seus filhos, netos e bisnetos. E o amor que me dedicou e a sabedoria de sua simplicidade sempre estarão comigo.

AGRADECIMENTOS

A realização desta dissertação é o resultado de dois anos de estudo e dedicação, muitas das coisas aprendidas não estão aqui inseridas, e este resultado só foi alcançado devido o apoio, incentivo e confiança de pessoas muito especiais para mim. Gostaria de agradecer em especial: ao meu orientador Enrique, pela dedicação e principalmente pela sua paciência durante o desenvolvimento deste trabalho. A meus pais Reginaldo e Rosa pelo constante apoio e incentivo incondicional. Ao meu amigo e parceiro Rafael. Aos amigos que dividiram apartamento comigo em algum período de tempo durante esses dois anos: Savana, Daniel, Priscila, Jacob e Bruno, obrigada pela paciência de vocês pessoal. Aos amigos antigos e sempre presentes Carol, Tutty, Luciana e Antônio, saibam que a amizade de vocês no decorrer desses quase dez anos é o que me ajuda a sempre querer ir mais longe e tentar me superar a cada dia. Aos meus companheiros de trabalhos e amigos: Paulo, Isis, Márcio, Romero, Cássia, Rogi e todo o pessoal do RISCTEC. E gostaria de agradecer de todo coração a amizade da Patrícia, Filipe e Sônia, o carinho de vocês me faz ser uma pessoa melhor, amo vocês. Muito Obrigada.

RESUMO

Na presente dissertação apresenta-se a modelagem, implementação e resultados de um sistema multiagentes baseado na metaheurística de colônia de formigas para resolver problemas multiobjetivos na área de engenharia de confiabilidade. Considera-se problemas de alocação de redundância e política de testes periódicos. Ao se projetar um sistema, deseja-se alta confiabilidade e que os custos associados sejam os menores possíveis. Entretanto, em geral, esses objetivos são conflitantes, inviabilizando encontrar uma única solução que obtenha ótimo desempenho para ambos. Porém, adotando-se uma abordagem multiobjetivo, pode-se encontrar um conjunto de soluções ótimas que satisfaçam as restrições do sistema. O aumento da confiabilidade de um sistema pode ser obtido, entre outras formas, pela alocação de redundâncias. Esse aumento, porém, incorre em maiores custos. Logo, deseja-se encontrar o número ótimo de equipamentos para cada um dos subsistemas, de modo a satisfazer restrições de custo e confiabilidade. A obtenção de intervalos de testes periódicos vem sendo muito discutida nos trabalhos recentes. Busca-se intervalos de testes que garantam altos níveis de confiabilidade, porém alta frequência de manutenção representa altos custos, nem sempre justificáveis pelo aumento da confiabilidade que possa proporcionar. Nos exemplos de aplicação são considerados tanto sistemas não-reparáveis, que não são passíveis de manutenção, quanto sistemas reparáveis, que após a falha voltam à operação por um procedimento que não seja a sua completa substituição. São resolvidos exemplos de aplicação, tanto para problemas de alocação de redundância, quanto de obtenção de intervalos de testes periódicos para sistemas de segurança de uma planta nuclear.

ABSTRACT

This dissertation presents the modeling, implementation and results of a multiagent system based on ant colony metaheuristic to solve multiobjective problems on reliability engineering area. We will consider redundancy allocation and periodic tests policy problems. High reliability and low associated costs are objectives of all systems projects. Meanwhile, in general, these are opposite objectives. So, find an optimal solution for both, is a impossible task. In fact, adopting a multiobjective approach we may find a group of optimal solutions that complains the system constraints. One way, between many others, to raise a system reliability is by redundancy allocation. But this increase will cause higher costs. So, the objective is to find the optimal equipment number for each subsystems, in way to satisfy the costs and reliability constraints. The way to get the periodic tests interval had been so much discussed on recent academic works. The search is for a high reliability level that is reached by applying a certain interval of tests but if the test frequency is so high, it will increase a lot the costs, and that could not compensate the reliability raise associated. In application examples, are consider both no repairable systems, that are no possibility to maintain, and repairable systems, that after a breakdown may return to normal operation by a procedure that is not a complete substitution. There are application examples for redundancy allocation problems and a periodic tests policy problem for security systems of a nuclear power plant.

SUMÁRIO

<i>AGRADECIMENTOS</i>	<i>iv</i>
<i>RESUMO</i>	<i>v</i>
<i>ABSTRACT</i>	<i>vi</i>
<i>SUMÁRIO</i>	<i>vii</i>
<i>LISTA DE FIGURAS</i>	<i>ix</i>
<i>LISTA DE TABELAS</i>	<i>xi</i>
1 INTRODUÇÃO	1
1.1 Contextualização	1
1.2 Justificativa	4
1.3 Objetivo	5
2 ENGENHARIA DE CONFIABILIDADE	7
2.1 Definições gerais	7
2.2 Configuração de sistemas	9
2.3 Tipos de sistemas	12
2.3.1 Sistemas não-reparáveis	12
2.3.2 Sistemas reparáveis	14
2.4 Análise da disponibilidade de sistemas	15
3 METAHEURÍSTICAS E OTIMIZAÇÃO MULTIOBJETIVO	19
3.1 Heurísticas e metaheurísticas	19
3.2 Otimização multiobjetivo	21
3.2.1 Problema de otimização multiobjetivo.....	21
3.2.2 Classificação dos métodos multiobjetivos	23
3.3 Otimização combinatória	24
3.4 Teoria dos grafos	25
4 SISTEMAS MULTIAGENTES	27
4.1 Computação natural	27
4.2 Inteligência artificial	29

4.3	Inteligência artificial distribuída	29
4.4	Sistemas multiagentes	30
4.4.1	Sistemas multiagentes reativos.....	31
4.4.2	Sistemas multiagentes cognitivos.....	32
5	<i>COLÔNIA DE FORMIGAS</i>.....	34
5.1	Definições gerais	34
5.2	Comportamento das formigas.....	35
5.3	ACO Combinatório.....	38
6	<i>METODOLOGIA PROPOSTA E IMPLEMENTAÇÃO</i>	40
6.1	Metodologia proposta	40
6.2	Implementação	44
7	<i>RESULTADOS</i>.....	48
7.1	Modelo da ponte dupla estendida	48
7.2	Problema de alocação de redundâncias	50
7.2.1	Formulação do problema.....	50
7.2.2	Exemplo de aplicação	52
7.3	Problema de otimização de uma política de testes periódicos.....	55
7.3.1	Resolução	56
7.3.2	Exemplo de validação	57
7.3.3	Definição do sistema auxiliar de água de alimentação.....	58
7.3.4	Cálculo do custo e da confiabilidade.....	59
8	<i>CONCLUSÕES E PERSPECTIVAS FUTURAS</i>.....	62
8.1	Conclusões.....	62
8.2	Limitações.....	63
8.3	Perspectivas futuras.....	63
	<i>REFERÊNCIAS BIBLIOGRÁFICAS</i>.....	64

LISTA DE FIGURAS

Figura 2.1 – Sistema em série.....	10
Figura 2.2 – Sistema em paralelo	10
Figura 2.3 - Redundância passiva.....	11
Figura 2.4 – Sistema série-paralelo.....	11
Figura 2.5 – Sistema em paralelo-série.....	12
Figura 2.6 - Função confiabilidade	13
Figura 2.7 - Curva da banheira.....	14
Figura 2.8 - Disponibilidade como parâmetro de projeto.....	16
Figura 2.9 - Região economicamente aceita.....	17
Figura 3.1- Princípio básico da metaheurística.....	19
Figura 3.2 - Métodos multicritério e suas subdivisões	21
Figura 3.3 - Dominância de Pareto no espaço objetivo.....	23
Figura 3.4 - Um grafo com 6 vértices e 7 arestas.....	26
Figura 4.1 - Integração de linhas de pesquisa para o desenvolvimento da computação natural.....	28
Figura 4.2 - Desenvolvimento ideal de sistemas multiagentes. Fonte: (HÜBNER, 2003).....	31
Figura 4.3 - Agente reativo.....	32
Figura 4.4 - Agente cognitivo.....	32
Figura 6.1 – Diagrama de um agente formiga se movimentando em um grafo.....	40
Figura 6.2 – Mecanismo de eliminação de ciclos.....	43
Figura 6.3 – Diagrama de pacotes	45
Figura 6.4 – Diagrama de classe do pacote graph	45
Figura 6.5 – Método construtor classe Edge	45
Figura 6.6 – Diagrama de classe do ambiente do sistema.....	46
Figura 6.7 – Método way da classe Ant.....	47
Figura 6.8 – Método depositPheromone da classe Ant.....	47
Figura 7.1 - Modelo da ponte dupla estendida.....	48
Figura 7.2- Desempenho do programa ACO-Multiagente.....	49
Figura 7.3 - Desempenho dos programas AntSim e Camponotus.....	50
Figura 7.4 - Sistema série-paralelo.....	51

Figura 7.5 – Comparação entre ACO-Multiagente e ACSRAP	54
Figura 7.6 - Fronteira de soluções não-dominadas para o exemplo RAP.....	55
Figura 7.7 – Lista de componentes Adaptado: (Samrout, 2005).....	56
Figura 7.8 – Exemplo de construção de solução (1 formiga e 5 componentes).....	56
Figura 7.9 – Estrutura do sistema em paralelo-série.....	57
Figura 7.10 – Diagrama simplificado do AFWS.....	59
Figura 7.11 – Resultado da otimização multiobjetivo para o problema AFWS.....	61

LISTA DE TABELAS

Tabela 6.1 – Analogias utilizados na modelagem do sistema.....	41
Tabela 6.2 – Atuação do agente-formiga de deposição de feromônio.....	42
Tabela 6.3 - Atuação do agente-formiga de acordo com o valor do limiar	42
Tabela 7.1 – Dados de entrada do problema de alocação de redundância.....	53
Tabela 7.2 – Parâmetros utilizados no sistema multiobjetivo	53
Tabela 7.3 – Resultados obtidos usando ACSRAP e ACO-Multiagentes	54
Tabela 7.4 – Vetor Tp obtido com os melhores custos usando ACS e ACO-Multiagentes	58
Tabela 7.5 – Tempos de parada para os componentes do sistema AFWS (h)	60
Tabela 7.6 – Parâmetros utilizados para o problema AFWS	60
Tabela 7.7 – Melhor política de testes encontrada	61

1 INTRODUÇÃO

“There is nothing more difficult to take in hand, more perilous to conduct, or more uncertain in its success, than to take the lead in the introduction of a new order of things.”

Niccolo Machiavelli, The Prince¹

No presente capítulo são apresentados: o contexto no qual este trabalho se insere, as justificativas para o desenvolvimento do estudo e os objetivos alcançados.

1.1 Contextualização

A busca pela qualidade e satisfação dos clientes demanda processos e serviços cada vez mais confiáveis. Para tanto, na fase de projeto do sistema, deve-se considerar não só as características inerentes ao processo produtivo, mas também a confiabilidade do sistema e os custos associados. Segundo Lewis (1996), confiabilidade é definida como a probabilidade de um componente ou sistema realizar a função pretendida, sob condições de operação pré-determinadas, e num certo período de tempo.

O conceito de confiabilidade passou a ser mais difundido após a Primeira Guerra Mundial com o crescente desenvolvimento da indústria aeronáutica. Na década de 40, houve o desenvolvimento de teorias matemáticas relacionadas ao problema. De acordo com Lafraia (2001), durante a Segunda Guerra Mundial, a falta de equipamentos eletrônicos da Coréia fomentou grande interesse militar dos Estados Unidos em confiabilidade, porém o grande salto nessa área ocorreu na década de 50, com o surgimento da indústria aeroespacial, eletrônica e nuclear. Na década de 70, com os conceitos consolidados surgem os primeiros modelos de análise de confiabilidade em programas computacionais. De acordo com as funções requeridas para sistemas de projetos práticos, pode-se subdividi-lo em um número específico de subsistemas. Para cada subsistema tem-se diferentes tipos de componentes disponíveis com diversas características como confiabilidades e custo. A confiabilidade do sistema depende da confiabilidade de cada subsistema.

O alcance de processos e serviços mais confiáveis pode ser obtido pela otimização da confiabilidade. Problemas de otimização são caracterizados por situações em que se deseja maximizar ou minimizar uma função numérica de várias variáveis, num contexto em que

¹ “The Prince” foi escrito em 1512 por Niccolo Machiavelli, um filósofo político italiano, que viveu entre 1469-1527.

podem existir restrições (SARAMAGO; PRADO, 2005). Para maximizar a confiabilidade, as seguintes medidas podem ser efetuadas: usar componentes mais confiáveis, usar configuração redundante ativa ou em modo de espera (*stand-by*). Neste caso, pode-se definir redundâncias como equipamentos adicionados ao sistema para aumentar a confiabilidade do mesmo. Kuo *et al.* (2001), classifica os problemas de otimização envolvendo projeto de sistemas baseados em confiabilidade em:

- Problemas de alocação de redundâncias, em que as variáveis de decisão representam o número de redundâncias;
- Problemas de alocação de confiabilidade, em que as variáveis de decisão são as confiabilidades dos componentes que constituem o sistema;
- Problemas de alocação de confiabilidade e redundâncias, em que as variáveis de decisão representam a confiabilidade dos componentes e o número de redundâncias;
- Problemas de atribuição de componentes, quando o arranjo dos componentes no sistema faz diferença na confiabilidade dos sistemas.

A redundância de componentes em sistemas é uma das formas de incrementar a confiabilidade, pois na falha de um componente, a unidade redundante passa a executar o objetivo de projeto. Entretanto, um sistema mais confiável é mais caro, pois o custo de projeto cresce muito com o aumento da confiabilidade. Os objetivos de projetar sistemas de alta confiabilidade e baixos custos são conflitantes. Porém, esses dois objetivos devem ser considerados de maneira conjunta no momento de elaboração do projeto, visando obter soluções satisfatórias para ambos.

De acordo com Rigdon e Basu (2000) e Lafraia (2001), sistemas reparáveis são aqueles passíveis de substituição parcial ou reparo, ou seja, são colocados em operação após a falha para realizarem a função requerida através de qualquer procedimento que não seja a completa substituição do mesmo. Se os tempos de reparo de tal sistema não são desprezíveis em relação ao tempo de operação, então a confiabilidade do mesmo é medida pela sua disponibilidade. Os sistemas não-reparáveis, por sua vez, são descartados após a falha, ou seja, eles não são passíveis de manutenção.

Rausand e Hoyland (2003) definem disponibilidade como a probabilidade do sistema encontrar-se operacional em um dado instante de tempo. Quando o sistema está em modo de espera, realizam-se testes periódicos para verificar se os componentes do mesmo estão ou não operacionais, revelando, dessa forma, possíveis falhas desses componentes.

Confiabilidade e disponibilidade são alguns dos parâmetros para avaliar a performance do sistema. Seus valores dependem da confiabilidade e disponibilidade dos componentes do mesmo. Em geral, estes valores podem vir a diminuir conforme a idade dos componentes aumenta, e são influenciados por políticas de manutenção. Entre os diferentes tipos de políticas de manutenção, a manutenção preventiva vem sendo muito aplicada em sistemas de transporte, produção, etc. A manutenção preventiva consiste de um conjunto de ações técnicas, administrativas e gerenciais para diminuir a idade dos componentes, melhorando desta forma, a confiabilidade e disponibilidade do sistema. Estas ações podem ser caracterizadas por seus efeitos na idade do componente que se torna: “tão bom quanto novo”, quando seu estado volta a condição inicial de operação; “tão ruim quanto velho”², volta a condição imediatamente anterior a manutenção.

Neste contexto, há diversos trabalhos utilizando técnicas como algoritmo genético que otimizam um único objetivo, tais como Legat *et al.* (1996) que determinou o intervalo ótimo para manutenção preventiva baseada na estratégia de renovação. Algoritmo genético (AG) foi utilizado por Levitin e Lisnianski (2000) para calcular o custo mínimo de uma política de manutenção preventiva. Tsai *et al.* (2001) usou AG para decidir uma combinação de atividade ótima que maximize a unidade do custo de vida do sistema. Bris *et al.* (2003), desenvolveu um novo método baseado em AG para minimizar o custo da manutenção preventiva de um sistema série-paralelo, a partir deste método Samrout *et al.* (2005) utilizou outra técnica baseada em colônia de formigas para determinar o vetor de soluções de inspeções periódicas de componentes do sistema.

Na tomada de decisão, vários aspectos (econômicos, sociais, políticos, ambientais, etc) devem ser considerados simultaneamente. Dificilmente uma decisão é tomada em função de um único objetivo. A otimização multiobjetivo surgiu na seqüência do desenvolvimento dos trabalhos com otimização. No estudo de diversos objetivos operacionais que os sistemas devem atender, verificou-se que uma formulação mais abrangente necessitaria ser desenvolvida e assim ocorreram muitos avanços no campo da otimização multiobjetivo³ (FRANCATO; BARBOSA, 2004). Com a utilização de técnicas multiobjetivo pode-se encontrar um conjunto de soluções potenciais que são igualmente adequadas do ponto de vista multiobjetivo (COELLO *et al.*, 2002). A meta é encontrar uma solução que forneça o melhor compromisso entre vários objetivos.

² Os principais processos de contagem usados em confiabilidade serão mostrados na subseção 2.5.1

³ Otimização multiobjetivo será abordada na seção 3.2

Há várias técnicas presentes na literatura utilizadas para a solução de problemas multiobjetivos. No contexto dos problemas que serão resolvidos neste trabalho, pode-se citar: Marseguerra *et al.* (2000), que utiliza algoritmo genético na otimização multiobjetivo da política de testes periódicos para uma planta de potência nuclear (NPP) do tipo “*Pressurized Water Reactor - PWR*”. O artigo trabalha com três funções objetivos (média da indisponibilidade, função custo e tempo de exposição), porém não utiliza restrições. Em Marseguerra *et al.* (2005), explora-se a possibilidade de usar algoritmo genético para a tarefa de otimizar o número de peças sobressalentes requeridas por um sistema com componentes múltiplos em NPPs. No artigo busca-se a maximização do lucro e a minimização do volume total de sobressalentes. Lapa *et al.* (2006) apresenta um modelo de manutenção preventiva por algoritmo genético baseado em custo e confiabilidade em uma planta nuclear PWR. Marseguerra e Zio (2000), apresentam a solução para um problema de otimização da manutenção e reparo periódico através da combinação de algoritmo genético e simulação de Monte Carlo.

Para problemas de alocação de redundância, Marseguerra *et al.* (2006), desenvolveu um algoritmo de otimização baseado em algoritmo genético. Taboada e Coit (2007) utilizaram algoritmo genético para maximização da disponibilidade e minimização do custo e peso do sistema. Zhao *et al.* (2007), por sua vez, baseado na heurística de colônia de formigas, também apresenta bons resultados para problemas de alocação de redundância.

1.2 Justificativa

Problemas de otimização da confiabilidade e projetos de sistemas envolvem vários critérios ou objetivos conflitantes, como, por exemplo, minimizar o custo e maximizar a confiabilidade. De acordo com Levitin (2007), nas últimas duas décadas, metaheurísticas⁴ têm sido a principal ferramenta para resolver problemas de otimização combinatória multiobjetivo, por apresentarem bons resultados para problemas combinatórios complexos dentro de um tempo razoável. Dentre as metaheurísticas existentes, destaca-se colônia de formigas, inicialmente proposto por Dorigo e Stützle (2004), que vem apresentando bons resultados para problemas de otimização combinatória. Nesse trabalho desenvolveu-se um sistema multiagentes⁵ baseado em colônia de formigas para a solução de problemas combinatórios. Como exemplos de aplicação utilizou-se o problema de alocação de

⁴ O conceito de metaheurísticas será dado na seção 3.1

⁵ Sistemas multiagentes será explicado no capítulo 4

redundâncias e otimização de uma política de testes periódicos para um sistema auxiliar de água de alimentação de uma típica planta de potência nuclear do tipo PWR de dois ciclos. O sistema desenvolvido é validado através da comparação dos seus resultados com os de Dorigo e Stützle (2004) para a solução do modelo da ponte dupla estendida. Os resultados da solução do problema de alocação de redundância são comparados com trabalhos similares, visando analisar o desempenho do sistema desenvolvido.

Em muitos projetos de sistemas é necessário dividi-lo em um número específico de subsistemas, de acordo com a função requerida pelo sistema. Para cada subsistema têm-se diferentes tipos de componentes, cada qual com diferentes características de disponibilidade, custo, confiabilidade, entre outros. A confiabilidade do sistema depende da confiabilidade de cada subsistema, e uma das formas de otimizar essa confiabilidade é identificar a combinação ótima para os tipos de componentes e o nível de redundância de cada subsistema. O problema se torna complexo à medida que simultaneamente deseja-se otimizar mais de um objetivo, como maximizar a confiabilidade e minimizar o custo, sujeitos a um conjunto de restrições como o número de componentes disponíveis.

Em uma típica planta de potência nuclear (“*Nuclear Power Plant*” - NPP) do tipo PWR (“*Pressurized Water Reactor*”), a política de manutenção aplicada para um sistema elétrico-mecânico obriga um alto nível de confiabilidade para seus componentes. Uma grande frequência de intervenções de manutenção pode representar custos desnecessários, que muitas vezes não correspondem ao aumento da confiabilidade dos componentes. De acordo com Duffey (2000) em uma planta de potência nuclear PWR, os custos de manutenção durante o tempo de vida representa cerca de 30% dos custos totais da NPP. Logo, pequena melhora na política de manutenção pode proporcionar um significativo ganho econômico. Neste trabalho, propõe-se uma nova metaheurística baseada em colônia de formigas para resolver o problema de teste periódico.

1.3 Objetivo

O objetivo mais geral desta produção é desenvolver um trabalho de pesquisa no campo de algoritmos de colônia de formigas para resolver problemas multiobjetivos na área de engenharia da confiabilidade, tendo como objetivos específicos os seguintes itens:

- Estudar conceitos de engenharia da confiabilidade, tais como, disponibilidade, configurações de sistemas e sistemas reparáveis, visando modelar os problemas de alocação de redundâncias e otimização de uma política de testes periódicos;

- Compreender a teoria de sistemas multiagentes;
- Estudar os problemas de otimização, com a intenção de entender o porquê de se aplicar algoritmos de formigas para resolvê-los;
- Estudar como as formigas agem na natureza, para poder modelar seu comportamento artificialmente;
- Modelar e implementar um sistema multiagentes capaz de resolver problemas multiobjetivo;
- Resolver através do sistema implementado, problemas de alocação de redundâncias e otimização de testes periódicos.

2 ENGENHARIA DE CONFIABILIDADE

“Originalidade é a arte de ocultar as fontes...”

*Benjamin Franklin*⁶

No presente capítulo apresenta-se a fundamentação teórica básica referente à engenharia de confiabilidade que é utilizada para a modelagem dos problemas tratados.

2.1 Definições gerais

De acordo com Lewis (1996), a importância da confiabilidade vem crescendo motivada por diversos fatores como, por exemplo: aumento da complexidade e sofisticação dos sistemas; conscientização do consumidor, e posterior exigência, com relação à importância da qualidade do produto; surgimento de leis e regulamentações estabelecendo responsabilidade do fabricante com relação ao seu produto; pressões econômicas resultantes de altos custos das falhas, reparos e programas de garantia. Porém, alguns fatores são limitantes para desenvolvimento desses sistemas perfeitos, tais como: elevados custos de desenvolvimento, materiais, testes, entre outras etapas do projeto o que tornam economicamente inviável a construção desse sistema; e engenheiros de projeto que não têm conhecimento total das condições de trabalho, de produção e manutenção de tais componentes. Devido a essas limitações econômicas e práticas, componentes e sistemas não são perfeitos, ocasionando uma probabilidade de falha durante seu tempo de vida. Estas falhas, no sistema ou de produtos, ocasionam impacto social e econômico.

Um volume substancial de conhecimento sobre a natureza estocástica da falha e formas de minimizar a probabilidade da mesma ocorrer foi desenvolvido. Dentre algumas técnicas para caracterizar a confiabilidade de sistemas, pode-se ressaltar o tempo médio de falha e a taxa de falhas. No caso de sistemas reparáveis, a disponibilidade e o tempo médio de reparo. Entende-se por falha, o fato de um sistema não poder mais cumprir sua função de projeto.

Os conceitos que serviram de base para o desenvolvimento do presente trabalho são explorados nas obras de Lewis (1996), Rigdon e Basu (2000), Modarres (1999) e Trivedi

⁶ Benjamin Franklin (1706 — 1790) foi um jornalista, editor, autor, filantropo, abolicionista, funcionário público, cientista, diplomata, um dos líderes da Revolução Americana, muito conhecido pelas suas muitas citações e experiências com a eletricidade.

(2002). Dependabilidade, definida por Wohl (1966), apresenta uma importante relação custo-benefício entre a manutenção e o tempo de funcionamento dos sistemas. A capacidade de recolocação em funcionamento de um componente ou sistema após uma falha pode ser quantificada por outra grandeza probabilística, a manutenibilidade. Aliando confiabilidade e manutenibilidade, pode-se determinar a probabilidade de que tal sistema esteja pronto para funcionamento em um dado instante, ou seja, o quão disponível o sistema será para uso. Esta grandeza é conhecida como disponibilidade (LEWIS, 1996).

O aumento da confiabilidade pode ser obtido considerando-se: o incremento da confiabilidade dos componentes do sistema; a alocação dos componentes redundantes em paralelo; a utilização de componentes intercambiáveis ou um hibridismo entre uma das opções anteriores. O aumento da confiabilidade dos componentes do sistema foi bastante explorado até 1970. O trabalho de Tillman *et al.* (1977) apresenta uma revisão dos trabalhos publicados até aquela data sobre otimização da confiabilidade em sistemas redundantes.

Sasaki (1977) propôs um método de maximização da disponibilidade de sistemas redundantes com custo e peso como restrições. Narasimhalu e Sivaramakrishnan (1978), propuseram um método eficiente para a otimização da confiabilidade em sistemas redundantes com várias restrições não necessariamente lineares. Segundo Castro (2003), nas contribuições feitas para otimização baseada na confiabilidade de sistemas de 1977 a 2000 foi dado maior atenção no desenvolvimento de métodos heurísticos e metaheurísticos. Os métodos de otimização do período analisado foram divididos em sete categorias:

- Métodos heurísticos;
- Algoritmos metaheurísticos;
- Algoritmos exatos;
- Métodos heurísticos para otimização da confiabilidade dos componentes;
- Otimização multicritério de sistemas;
- Otimização de componentes intercambiáveis;
- Outros métodos.

O aumento da confiabilidade pode ser obtido de várias formas e a escolha da forma depende da natureza do equipamento, do seu custo e da sua missão. Também se pode maximizar a confiabilidade através da otimização da manutenibilidade e disponibilidade, além de se minimizar os recursos consumidos, como o custo, e algumas características físicas do sistema como a massa e o volume. Nesses casos, métodos de otimização com objetivos múltiplos são necessários para a solução destes problemas.

Lafraia (2001) ressalta alguns benefícios na aplicação da confiabilidade, como fornecer soluções às necessidades atuais das indústrias permitindo a aplicação de investimentos com base em informações quantitativas, segurança e meio ambiente; eliminar as causas básicas de paradas não programadas de indústrias ou instalação atuando nas causas e não nos sintomas, para tanto, utiliza-se o histórico de falhas dos equipamentos. Além destes, há tantos outros benefícios, ressaltando-se a prevenção de falhas em equipamentos similares e a determinação de fatores críticos para manutenibilidade de equipamentos.

Ressalta-se duas distribuições de probabilidade freqüentemente usadas para descrever o processo de falhas dos componentes: exponencial, que possui taxas de falha e reparo constantes, adequando-se a equipamentos com longa vida útil (eletrônicos); e *Weibull*, que simula os efeitos de desgaste e fadiga, inerentes aos sistemas mecânicos. A variação do parâmetro de forma β faz com que a distribuição de *Weibull* se ajuste a vários modelos físicos. Para um estudo mais completo, recomenda-se Lewis (1996), Wackerly (1996) e Rigdon e Basu (2000).

2.2 Configuração de sistemas

Em engenharia de confiabilidade, quando a complexidade do sistema aumenta, a confiabilidade tende a diminuir. Freqüentemente, utiliza-se o número de componentes não redundantes do sistema para medir a complexidade do mesmo. Se as taxas de falhas são mutuamente independentes, a confiabilidade do sistema com n componentes não redundantes é dada por $R = R_1 R_2 \dots R_n$, ou seja, a confiabilidade diminui com n . Uma alternativa para o aumento de componentes, o que diminui a confiabilidade, é prover redundância em parte ou em todo sistema. A utilização de redundância dos componentes gera aumento da confiabilidade, bem como o aumento do custo do sistema. Na forma simples de redundância do sistema, a configuração dos componentes está em paralelo.

Segundo Modarres (2006), o arranjo físico dos componentes no sistema pode ser representado por meio de um diagrama de blocos. Com o diagrama de blocos, é possível identificar quais componentes devem funcionar para que o sistema como um todo também funcione, ou seja, identificar os seus caminhos mínimos. Considerando-se para efeito de simplificação os valores para as confiabilidades dos componentes constantes, um sistema formado por n componentes, onde $P[O_i]$, $i = 1, 2, 3, \dots, n$, é a probabilidade de ocorrência do

evento O_i em que i opera satisfatoriamente por um período de tempo pré-determinado e $\overline{O_i}$ é o evento complementar de O_i .

Um sistema em série, Figura 2.1, só está operacional se todos os seus componentes estão operacionais e sua confiabilidade é calculada da seguinte forma:

$$R_S = P[O_1 \cap O_2 \cap \dots \cap O_n] = \prod_{i=1}^n P[O_i] \quad (2.1)$$

$$R_S = \prod_{i=1}^n r_i$$

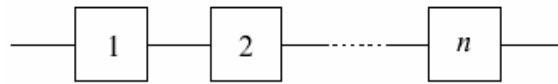


Figura 2.1 – Sistema em série

Na configuração em paralelo (ou configuração redundante) é necessário que ao menos um dos componentes esteja operacional para o sistema ser considerado operacional (ver Figura 2.2). Pode-se incrementar a confiabilidade do sistema pela alocação de n componentes em paralelo – procedimento também denominado de redundância múltipla. Para um sistema com n componentes em paralelo tem-se:

$$R_S = P[O_1 \cup O_2 \cup \dots \cup O_n] = 1 - \prod_{i=1}^n (1 - P[O_i]) \quad (2.2)$$

$$R_S = 1 - \prod_{i=1}^n (1 - r_i)$$

Se os componentes são idênticos, pode-se fazer uma simplificação:

$$R_S = 1 - (1 - r)^n \quad (2.3)$$

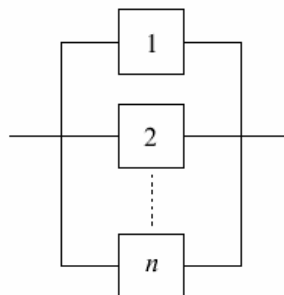


Figura 2.2 – Sistema em paralelo

As redundâncias podem ser ativas ou passivas. Uma redundância ativa constitui um conjunto de componentes em paralelo, onde o funcionamento dos mesmos não depende da falha de alguns deles. Neste caso, o sucesso da operação do sistema depende ao menos do sucesso de um de seus componentes. Já na redundância passiva, ou redundância em modo de espera (*standby*), é uma estrutura em paralelo, onde um dos componentes funciona após a falha do outro. O esquema mostrado na Figura 2.3 representa um sistema em modo de espera:

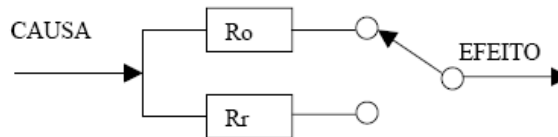


Figura 2.3 - Redundância passiva. Fonte: (CASTRO, 2003)

Em um sistema produtivo, pode-se encontrar outros tipos de configurações, tais como, série-paralelo e paralelo-série. A Figura 2.4 apresenta um exemplo de sistema série-paralelo formado por k subsistemas conectados em paralelo, em que cada subsistema i é composto por n_i componentes em série, da seguinte forma (LEWIS, 1996):

$$R_S = 1 - \prod_{i=1}^k (1 - \prod_{j=1}^{n_i} r_{ij}) \quad (2.4)$$

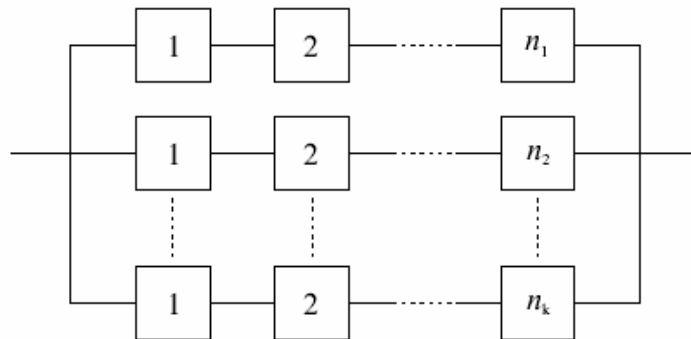


Figura 2.4 – Sistema série-paralelo

Se o sistema for composto por k subsistemas conectados em série, cada subsistema i formado por n_i componentes em paralelo, tem-se um sistema paralelo-série como ilustra a Figura 2.5, então:

$$R_S = \prod_{i=1}^k (1 - \prod_{j=1}^{n_i} r_{ij}) \quad (2.5)$$

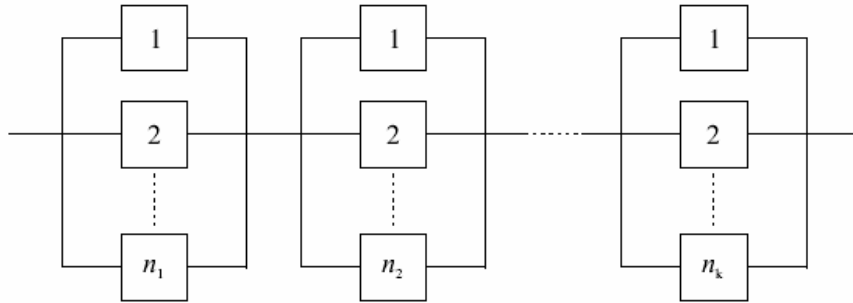


Figura 2.5 – Sistema em paralelo-série

Além das configurações série-paralelo e paralelo-série há outras como, série-paralelo hierárquicos e configurações complexas, que não serão abordadas neste trabalho, para maiores informações consultar (KURO *et al.*, 2001).

2.3 Tipos de sistemas

2.3.1 Sistemas não-reparáveis

Um sistema é não-reparável se, após sua primeira e única falha, o mesmo for descartado ou substituído por um novo, não sendo passível de manutenção. De acordo com Rigdon e Basu (2000), os tempos de operação até a falha são variáveis aleatórias independentes, além disso, os tempos de falha do sistema são identicamente distribuídos, ou seja, os tempos de falha são IID (independentes e identicamente distribuídos) com uma distribuição de probabilidade de função acumulada $F(x)$ para a variável aleatória X é definida a probabilidade que falhe antes x ,

$$F(x) = P(X \leq x) \quad (2.6)$$

A função de sobrevivência $S(x)$, também chamada de função de confiabilidade é a probabilidade que o sistema falhe (ou sobreviva) depois de x .

$$S(x) = P(X > x) = 1 - F(x) \quad (2.7)$$

A Figura 2.6 apresenta o comportamento da função confiabilidade no tempo.

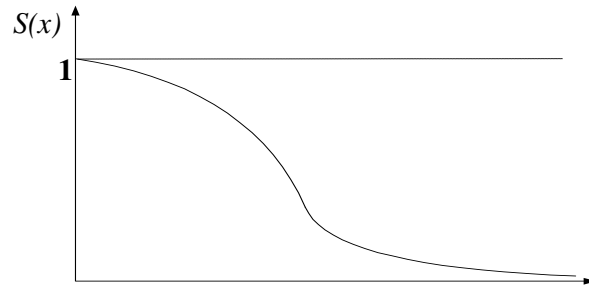


Figura 2.6 - Função confiabilidade

Define-se probabilidade de falha como a probabilidade de um dispositivo, ou sistema, falhar, ou deixar de desempenhar suas funções de projeto, em um período de tempo definido, sob certas condições operacionais. A probabilidade de falha $Q(x)$ é o complementar da confiabilidade e é dada por (RIGDON; BASU, 2000):

$$Q(x) = 1 - S(x) \quad (2.8)$$

A taxa de reparo μ é a razão entre o número de reparos e o tempo necessário para efetuar tais reparos. A função densidade de probabilidade $f(x)$ é a derivada da função da probabilidade de falha $Q(t)$.

$$f(x) = -\frac{d}{dx}S(x) = \frac{d}{dx}Q(x) \quad (2.9)$$

e a taxa de falha $h(x)$ é dada pela seguinte equação (RIGDON; BASU, 2000):

$$h(x) = \lim_{\Delta x \rightarrow 0} \frac{P(x < X \leq x + \Delta x | X > x)}{\Delta x} \quad (2.10)$$

A taxa de falha pode ser definida como o número de falha em um dado período de tempo. Uma maneira mais direta de se determinar $h(x)$:

$$h(x) = \frac{f(x)}{S(x)} \quad (2.11)$$

A análise do comportamento da taxa de falha de um equipamento por um longo período de tempo tem sido realizada por uma curva que possui a forma de uma banheira. Na Figura 2.7 percebe-se três regiões distintas. Na região I, há decrescimento da taxa de falha, também chamado de período de mortalidade infantil, as falhas diminuem porque os defeitos são reparados; na região II as falhas ocorrem de forma aleatória; e na região III devido a possíveis desgastes há crescimento do número de falhas, neste caso são necessárias intervenções de manutenção.

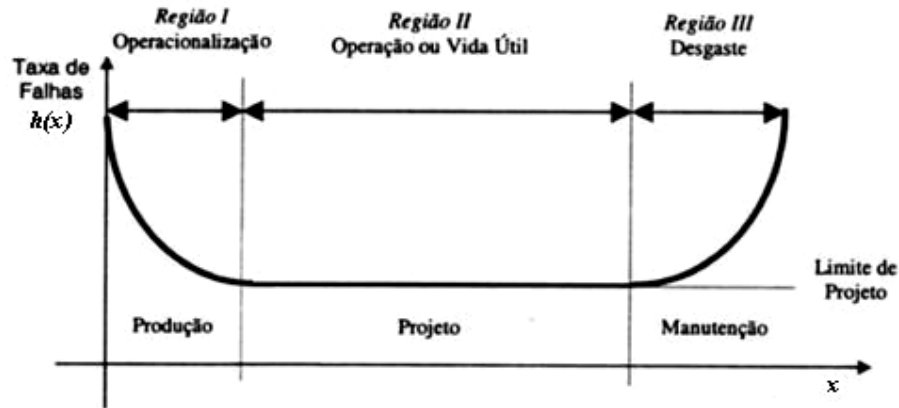


Figura 2.7 - Curva da banheira. Adaptado: (RIGDON; BASU, 2000)

2.3.2 Sistemas reparáveis

De acordo com Lewis (1996), os sistemas reparáveis são passíveis de substituição parcial ou reparo até a falha. Ressalta-se que a completa substituição de um componente é considerada um reparo. Há três modelos para sistemas reparáveis: substituição, manutenção e reparo.

O modelo de substituição é usado quando um item não-reparável é substituído por um outro após a falha, sendo que este item pode ser um sistema ou componente. Deve-se escolher a política de substituição a ser seguida quando da falha de um item. Modarres (1999) afirma que as políticas de substituição podem ser:

- **Substituição após a falha:** os itens são substituídos somente após falharem. Este modelo é apropriado se as falhas não são catastróficas e o custo associado com a substituição não é elevado, pois o tempo de detecção e substituição é considerado desprezível;
- **Substituição por idade:** os itens são substituídos após falharem ou após atingirem uma determinada idade c , o que acontecer primeiro. Neste caso, o tempo de substituição é desprezível.
- **Substituição em blocos:** os itens são substituídos após a falha e ao atingir os tempos operacionais c , $2c$, $3c$, etc. Onde c é um período pré-determinado no qual é conveniente substituir todos os itens operacionais. Esta política é mais fácil de administrar, porém pode resultar na substituição de um item que tem estado em operação por um curto período de tempo.

A escolha do modelo de substituição apropriado depende da distribuição do tempo até falhar do item e dos custos de falhas e administrativos.

No modelo de manutenção, a manutenção pode ser preventiva ou corretiva. Segundo Lewis (1996), o primeiro critério para julgar a manutenção preventiva é a confiabilidade. O critério mais utilizado em um sistema de manutenção é a probabilidade de o sistema estar operacional quando necessário. Na manutenção preventiva há realização de atividades no sistema antes da falha. Na manutenção corretiva as ações são realizadas no sistema após falha do mesmo. Ambas as manutenções preventiva e corretiva acarretam na retirada de operação do sistema por um período de tempo.

No modelo de reparo, os sistemas podem apresentar falhas reveladas ou não. Quando as falhas são reveladas o reparo pode iniciar imediatamente. Neste caso duas grandezas interessam: o número superior de falhas e a disponibilidade do sistema. O número de falhas é necessário para o cálculo dos custos de reparo. A disponibilidade é estimada através de dois modelos: a taxa de reparo constante e o tempo de reparo constante.

Quando o sistema não tem operação contínua, as falhas podem ocorrer, mas permanecem desconhecidas. Este problema é freqüente em *backup* ou outros equipamentos de emergência que operam somente raramente. A primeira perda de disponibilidade pode ocorrer por falhas em modo de espera (*standby*) que não foram detectadas até uma tentativa de utilizar o sistema. Uma forma de prevenir essa classe de falhas são os testes periódicos.

2.4 Análise da disponibilidade de sistemas

Segundo Rausand e Hoyland (2003), há quatro métricas de disponibilidade: instantânea, limite, média e média limite. A disponibilidade instantânea, $A(t)$, é definida como sendo a probabilidade de um sistema estar disponível para o uso no tempo t . Sendo $E[X(t)]$ o valor esperado da variável aleatória $X(t)$, então,

$$A(t) = P[X(t) = 1] = E[X(t)], \quad t \geq 0, \quad (2.12)$$

A disponibilidade média de um sistema pode ser matematicamente definida como a porcentagem do tempo que o mesmo está operacional e considera tanto o tempo operacional do sistema (confiabilidade), quanto o tempo fora de serviço (*downtime* do sistema - manutenibilidade):

$$A = \frac{\text{tempoOperacional}}{\text{tempoOperacional} + \text{tempoForaDeServiço}} \quad (2.13)$$

$$A_{med}(0,T) = \frac{1}{T} \int_0^T A(t) dt \quad (2.14)$$

onde T representa o tempo de vida útil do produto ou o tempo de missão.

A disponibilidade limite é dada pela Equação 2.15 (RAUSAND; HOYLAND, 2003), e pode ser interpretada como a fração do tempo em que o item está disponível considerando um longo tempo.

$$A_{\infty} = \lim_{t \rightarrow \infty} A(t) \quad (2.15)$$

Já a disponibilidade média limite em um dado intervalo (0,t] é apresentada na Equação 2.15 (RAUSAND; HOYLAND, 2003), se houver disponibilidade limite, então a disponibilidade média limite será igual a disponibilidade limite.

$$A_{med,\infty} = \lim_{t \rightarrow \infty} A_{med}(t) \quad (2.16)$$

Em geral, o aumento da disponibilidade gera aumento no custo, ou seja, para sistemas que requerem alta confiabilidade e performance, o custo é elevado, logo o processo de otimização deve obter o máximo destes parâmetros dentro dos limites de custo determinados. Nos casos em que a disponibilidade e a performance não são pontos críticos, pode-se obter custos mínimos definindo-se limites inferiores de disponibilidade e performance. Essa relação entre disponibilidade e custo foi apresentada por Ertas (1993). Na Figura 2.8, verifica-se que a disponibilidade diminui com o aumento da performance e o custo aumenta com a disponibilidade.

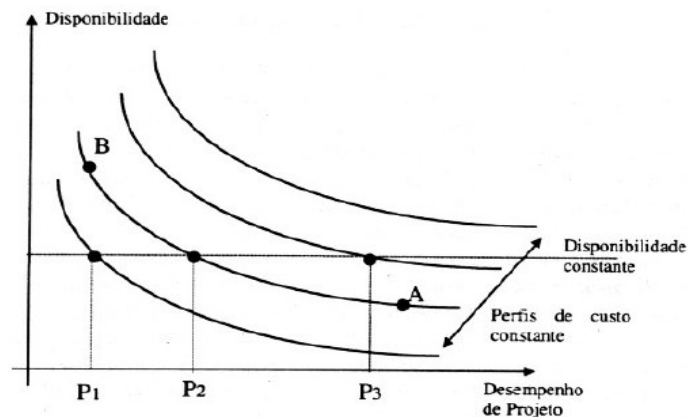


Figura 2.8 - Disponibilidade como parâmetro de projeto. Adaptado: (CASTRO, 2003)

A soma do tempo médio até a falha (*MTTF*) e o tempo médio até o reparo resulta no tempo médio entre falhas (*MTBF*). O *MTTF* é dado pela seguinte expressão (RAUSAND; HOYLAND, 2003):

$$MTTF = \int_0^{\infty} R(t)dt \quad (2.17)$$

Considerando-se Equação 2.13, determina-se uma relação linear entre o *MTTF* e o *MTTR*:

$$MTTR = \left(\frac{1-A}{A} \right) MTTF \quad (2.18)$$

Castro (2003) definiu um tempo mínimo para o *MTTF*, um valor máximo para o *MTTR*, e uma disponibilidade mínima permitida, e traçou a região economicamente aceita, onde o sistema poderá atuar, sem que os limites econômicos sejam quebrados (ver Figura 2.9).

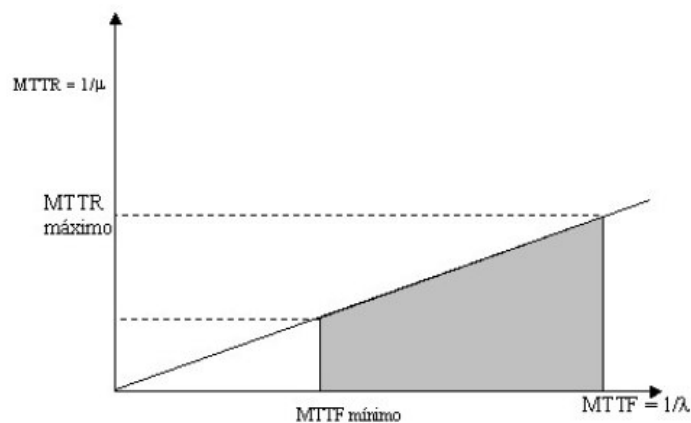


Figura 2.9 - Região economicamente aceita. Fonte: (CASTRO, 2003)

Para a confiabilidade é freqüente examinar a disponibilidade do sistema em termos da disponibilidade dos seus componentes. As análises podem prover compreensão dentro dos ganhos obtidos através da configuração de redundância, e diferentes estratégias de testes e reparo. Desde que a disponibilidade, como a confiabilidade, é uma probabilidade, a disponibilidade do sistema pode ser determinada pela combinação série e paralelo da disponibilidade de componentes.

Para sistemas sem redundância, a confiabilidade obedece a lei do produto:

\tilde{X} = estado de falha do sistema;

\tilde{X}_i = estado de falha do componente i ;

Em um sistema sem redundância a disponibilidade do sistema é dada por (RAUSAND; HOYLAND, 2003):

$$A(t) = \prod_i A_i(t) \quad (2.19)$$

Onde: $A_i(t)$ são as disponibilidades de componentes independentes;

3 METAHEURÍSTICAS E OTIMIZAÇÃO MULTIOBJETIVO

“In Theory, there is no difference between theory and practice. But in practice, there is a difference.”

Dorigo M. & Stützle T.⁷

No presente capítulo são apresentados de forma concisa conceitos referentes à metaheurísticas, otimização multiobjetivo e otimização combinatória, além de aspectos básicos da teoria dos grafos.

3.1 Heurísticas e metaheurísticas

Métodos de programação matemática tais como: programação linear, programação não-linear e programação inteira possuem aplicações nos mais diferentes campos de engenharia e das ciências. Porém, estes métodos podem apresentar algumas dificuldades numéricas quando as funções apresentam características como não diferenciabilidade, multimodalidade e espaço de busca complexos (KIRKPATRICK *et al.*, 1983).

A sofisticação dos recursos computacionais desenvolvidos nos últimos anos, bem como o aumento da complexidade dos problemas tem motivado um grande avanço nas técnicas de otimização. Assim, os estudos de métodos heurísticos, com busca randômica controlada por critérios probabilísticos, reaparecem como uma forte tendência, principalmente devido ao avanço dos recursos computacionais, pois um fator limitante destes métodos é a necessidade de um número elevado de avaliações da função objetivo (SCHWEFEL, 1994). Exemplos de metaheurísticas são: *simulated annealing* (SA) ou cozimento simulado, busca tabu, algoritmo genético, colônia de formigas e algoritmos meméticos. Apesar das diferenças técnicas metaheurísticas, essas podem ser descritas como um ciclo como mostra a Figura 3.1.

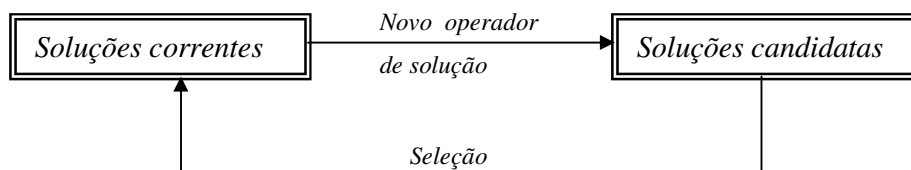


Figura 3.1- Princípio básico da metaheurística. Fonte: (LEVITIN, 2007, adaptação)

⁷ DORIGO, M.; STÜTZLE, T. *Ant Colony Optimization*. Cambridge, Massachusetts, 2004.

Segundo Dorigo e Stützle (2004), em respeito às heurísticas, abrangem programas monolíticos que, geralmente com resultados de otimalidade não tão boa, retornam uma ou mais soluções para um dado problema. Como exemplo, pode-se citar os algoritmos construtivos, que geram rascunhos de soluções através da adição iterativa de componentes, até que uma solução completa seja obtida; os algoritmos de busca local, que partem de uma solução inicial e tentam melhorá-la a cada iteração através de modificações locais; e os algoritmos exatos interrompidos na *n-ésima* iteração, compreendem os programas de busca exaustiva que não são executados exaustivamente.

Com relação às metaheurísticas, estas podem ser definidas como conjuntos de princípios algorítmicos que orientam a definição de métodos heurísticos aplicáveis a um vasto conjunto de problemas diferentes. Ou seja, uma metaheurística pode ser vista como um *framework* de algoritmo genérico aplicável em diferentes problemas de otimização, com modificações relativamente pequenas de modo a fazê-los adaptados a um problema específico.

Segundo Arroyo (2002), em otimização, a escolha do método de resolução a ser utilizado depende principalmente da razão entre a qualidade da solução gerada pelo método e o tempo gasto para encontrar essa solução. Nesse nível, a maioria dos problemas é intratável, ou seja, são problemas para os quais é improvável que se consiga desenvolver um algoritmo exato que possa ser executado em tempo razoável. Para viabilizar a obtenção de soluções é preciso lançar mão de métodos heurísticos. Esses métodos, quando bem desenvolvidos e adaptados aos problemas propostos, são capazes de apresentar soluções de boa qualidade em tempo compatível com a necessidade de rapidez presente nos problemas.

O desenvolvimento e sucesso dos métodos heurísticos, em especial as metaheurísticas, fomentou o interesse dos pesquisadores na década de 1990 na aplicação desses métodos em problemas de otimização combinatória multiobjetivo, considerados difíceis computacionalmente (EHRGOTT; GANDIBLEUX, 2000).

Nos métodos de otimização natural, a função objetivo é avaliada várias vezes, sendo possível trabalhar com vários pontos ao mesmo tempo em uma iteração (população). Isto eleva o custo computacional destes métodos. Entretanto, estes métodos possuem menor probabilidade de convergirem prematuramente. De forma geral, os métodos de otimização natural requerem maior esforço computacional quando comparados aos métodos de programação matemática, mas apresentam vantagens tais como: fácil implementação, robustez e não requerem continuidade na definição do problema. Como exemplo desta classe de métodos têm-se os algoritmos genéticos, que trabalham com técnicas de computação

evolutiva, as quais modelam a evolução das espécies proposta por Darwin e operando sobre uma população de candidatos (possíveis soluções). A idéia é que a evolução da população faça com que a formação dos cromossomos dos indivíduos caminhe para o ótimo, à medida que aumenta sua função de adaptação (*fitness*). Existem várias aplicações dos algoritmos genéticos em problemas de otimização multiobjetivo, consultar (BUSACCA *et al.*, 2001) e (FONSECA;FLEMING, 1993).

As metaheurísticas são métodos inteligentes e flexíveis, pois possuem uma estrutura com componentes genéricos que são adaptados ao problema que se quer resolver. Estes métodos possuem facilidade em incorporar novas situações e exploram o espaço de soluções permitindo a escolha estratégica de soluções melhores que as já encontradas, na tentativa de superar a otimalidade local. Mesmo não garantindo otimalidade global, as metaheurísticas podem encontrar uma grande quantidade de ótimos locais (ARROYO, 2002).

3.2 Otimização multiobjetivo

3.2.1 Problema de otimização multiobjetivo

Um problema de otimização pode possuir uma única função a ser minimizada ou maximizada. Nesta situação o ponto ótimo é simplesmente um máximo ou mínimo. Entretanto, pode-se considerar mais de um objetivo, comumente de natureza conflitante. Por exemplo, maximizar a confiabilidade ao mesmo tempo em que se deseja minimizar o custo de um dado sistema. Nestes casos, utiliza-se da abordagem multiobjetivo que de acordo com Yoon e Hwang (1995) é um sub-grupo dos métodos que auxiliam a tomada de decisão envolvendo múltiplos critérios (*Multiple Criteria Decision-Making*), ver Figura 3.2. Os métodos multicritério possuem um conjunto de critérios a serem julgados e um conjunto de variáveis de decisão, além de um processo de comparação das alternativas.

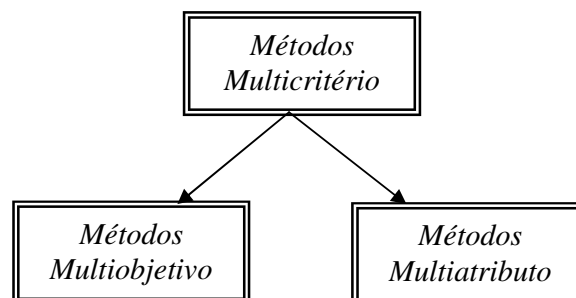


Figura 3.2 - Métodos multicritério e suas subdivisões

A otimização multiobjetivo pretende resolver problemas com objetivos múltiplos representados por funções f_1, f_2, \dots, f_m cujo domínio é um conjunto $\Omega \subseteq \mathbb{R}_n$. Basicamente, estabelece uma forma de tomar decisão que seja razoável aos objetivos mensurados. Em geral, não existem soluções ótimas no sentido de minimizarem (ou maximizarem) simultaneamente todos os objetivos, não possuem alternativas pré-determinadas, têm um conjunto de objetivos quantificáveis, um conjunto de restrições e um processo para obter informações sobre o compromisso entre os objetivos.

Os métodos multiatributo por sua vez, possuem um limitado número de alternativas pré-determinadas, suas variáveis de decisão não são necessariamente quantificáveis e a seleção da alternativa ocorre através de comparações inter e intra-atributos, considerando-se a estrutura de preferências do decisor (YOON; HWANG, 1995). Verifica-se que os métodos multiobjetivo são mais adequados para problemas relacionados à confiabilidade, pois a confiabilidade e o custo em projetos de sistema são variáveis quantificáveis. Além disso, a definição de um número pré-determinado de alternativas possibilita perda de possíveis combinações dos objetivos.

A característica principal da otimização multiobjetivo (quando todos os objetivos são de igual importância) é a existência de um grande conjunto de soluções não-dominadas. Estas soluções são também conhecidas como conjunto ótimo de Pareto ou ponto eficiente e foram introduzidas por Pareto onde um ponto é considerado Pareto eficiente quando não é possível melhorar nenhum objetivo sem piorar algum outro. Vilfredo Pareto generalizou a idéia de “ótimo” no contexto multiobjetivo (COELLO *et al.*, 2002). A escolha de uma solução eficiente particular depende das características próprias do problema e é atribuída ao decisor (*decision maker*). Representa-se um problema multiobjetivo da seguinte forma:

$$\text{Minimizar (ou maximizar): } Z = f(x) = [f_1(x), f_2(x), \dots, f_k(x)]$$

$$\text{Sujeito a: } g_i(x) = 0 \quad i = 1, \dots, j$$

$$h_i(x) \leq 0 \quad i = j+1, \dots, l$$

$$x = (x_1, x_2, \dots, x_n) \in X$$

$$z = (z_1, z_2, \dots, z_k) \in Z$$

Onde, \mathbf{x} é o vetor n -dimensional de variáveis de decisão, \mathbf{z} é o vetor formado por k funções-objetivo, X denota o espaço de decisões, $Z = f(X)$ é a imagem de X , j é o número de

restrições de igualdade e $l-j$ é o número de restrições de desigualdade. As restrições e o espaço de busca X determinam o conjunto das soluções factíveis X^* .

A imagem de X^* é denominada espaço objetivo factível e é denotada por $Z^* = f(X^*) = \{f(x) : x \in X^*\}$. Note que, a imagem de uma solução $x = (x_1, x_2, \dots, x_n) \in X^*$ no espaço objetivo é um ponto $z = (z_1, z_2, \dots, z_k) = f(x)$ tal que $z_j = f_j(x)$, $j = 1, \dots, k$. A Figura 3.3 apresenta o espaço objetivo factível para um dado problema de minimização com dois objetivos. O conjunto viável: (\underline{P}) será um conjunto ótimo de Pareto local se para todo $x \in \underline{P}$, não existe solução $x' \in X^*$ satisfazendo $\|x' - x\| < \varepsilon$ que domine qualquer membro do conjunto \underline{P} ($\|\cdot\|$ é distância entre os dois pontos e ε é um número positivo pequeno); e (\bar{P}) será um conjunto ótimo de Pareto global se para todo $x \in \bar{P}$, não existe $x' \in X^*$ tal que $x' \succ x$ (onde o sinal \succ indica dominância, x' é uma solução não-dominada e x é uma solução dominada para o mesmo problema). A fronteira global de Pareto é obtida quando se aplica as soluções pertencentes ao conjunto \bar{P} nas funções-objetivo.

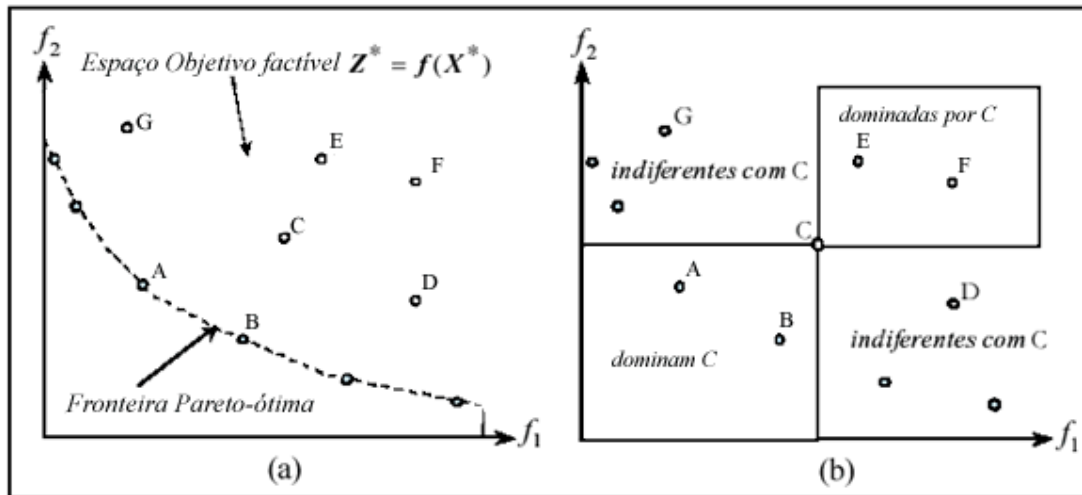


Figura 3.3 - Dominância de Pareto no espaço objetivo. Adaptado: (ARROYO, 2002)

3.2.2 Classificação dos métodos multiobjetivos

Segundo Zitzler (1999), há duas etapas para solução de problemas de otimização multiobjetivo: busca de soluções e tomada de decisões. A primeira etapa refere-se ao processo de otimização na qual a região factível é direcionada para soluções Pareto eficiente. A tomada de decisões envolve a seleção de um critério adequado para a escolha de uma solução do

conjunto ótimo de Pareto. Observadas essas etapas, classifica-se os métodos multiobjetivos em três categorias:

Métodos a-priori

A decisão ocorre antes da busca, há dois métodos a-priori: os objetivos são agregados em uma única função-objetivo; os objetivos são classificados em ordem decrescente de prioridade e resolvidos de forma independente na ordem em que foram classificados.

Métodos a-posteriori

Primeiro realiza-se a busca das soluções para posteriormente efetuar o processo de decisão. Considera-se que todos os objetivos são de igual importância. Ao final do processo da busca tem-se um conjunto ótimo de Pareto do qual seleciona-se a solução adequada para o problema.

Métodos iterativos

O decisor guia o processo de busca; a cada iteração elas são usadas para determinar novas soluções.

Ressaltam-se na próxima sessão alguns conceitos relevantes para o entendimento do problema e a forma de resolução desenvolvida neste trabalho.

3.3 Otimização combinatória

Os problemas de otimização combinatória (*Combinatorial optimization problems* - COPs) são oriundos de aplicações práticas, tais como, problemas de planejamento e programação (*scheduling*) da produção, problemas de corte e empacotamento, redes de telecomunicação, sistemas de distribuição de energia elétrica, problemas de localização, etc. Segundo Papadimitriou e Steiglitz (1982), um problema de otimização combinatória $P = (S, f)$ é um problema de otimização dado por um conjunto de soluções candidatas S e uma função objetivo $f: S \rightarrow \mathfrak{R}^+$ que atribui um valor de custo positivo para cada solução. O objetivo é encontrar a solução com o valor de custo mínimo ou, no caso de solução de técnicas aproximadas, uma solução boa o suficiente em um tempo aceitável.

A otimização combinatória encontra a combinação ou permutação ótima dos componentes disponíveis do problema. Esta classe de problemas apresenta domínio finito. Embora os elementos do domínio possam, em geral, ser facilmente enumerados (método chamado de busca exaustiva); a idéia de testar todos os elementos na busca pelo melhor deles mostra-se inviável para problemas com domínio muito grande. Os métodos de busca exaustiva são algoritmos, cuja complexidade é exponencial quando as possíveis soluções

crecem de forma exponencial conforme o aumento das variáveis de entrada do problema. Algoritmos cujo crescimento do número de operações básicas necessárias é polinomial, conforme o incremento das variáveis de entrada, são ditos de complexidade polinomial. De acordo com Dorigo e Stützle (2004), a complexidade de um problema pode ser mensurada através da complexidade do melhor algoritmo conhecido para resolvê-lo. Logo, se o melhor algoritmo que resolve um problema tem complexidade exponencial, tal problema é dito intratável. Caso contrário, a complexidade de tal algoritmo é polinomial e o problema é dito tratável. Os problemas podem ser divididos em duas classes de interesse: a classe P, que abrange problemas solucionáveis por algoritmos determinísticos polinomiais, e a classe NP, que compreende problemas solucionáveis por algoritmos não-determinísticos polinomiais.

De acordo com Dorigo e Stützle (2004), um problema para o qual é possível reduzir polinomialmente todos os problemas NP é dito NP-difícil. Se tal problema também for NP, então pode ser chamado NP-completo. Como exemplo de problema NP-completo tem-se o caixeiro viajante. Este problema é muito utilizado para avaliar a eficiência de algoritmos, como se verifica em Lawler *et al.* (1985), Souza (2003) e, Dorigo e Gambardellab (1997). Em geral, problemas de otimização combinatória são fáceis de serem formulados, mas difíceis de serem resolvidos. Muitos desses problemas pertencem à classe NP-difícil, ou seja, no pior caso, o esforço necessário para solucioná-los otimamente cresce exponencialmente com o seu tamanho.

Para a solução deste tipo de problema pode-se empregar algoritmos exatos ou aproximados. Os algoritmos exatos são procedimentos de busca exaustiva, apesar de garantir encontrar o ótimo, para problemas NP-difíceis seu desempenho não é satisfatório. Os algoritmos aproximados não garantem a solução ótima, porém obtém boas soluções a um custo computacional de tempo polinomial. Como exemplos de algoritmos aproximados têm as metaheurísticas: colônia de formigas, algoritmo genético e *simulated annealing*. Para maiores detalhes sobre otimização combinatória recomenda-se Cook *et al.* (1993), Korte e Vygen (2002) e, Dorigo e Stützle (2004). Ressalta-se o conceito referente à teoria dos grafos, devido sua importância para a metaheurísticas de otimização de colônia de formigas.

3.4 Teoria dos grafos

A teoria dos grafos é o ramo da matemática que estuda as propriedades de grafos. Um grafo é um conjunto de pontos, chamados vértices, conectados por linhas, chamadas de arestas. Um exemplo de grafo pode ser visto na Figura 3.4.

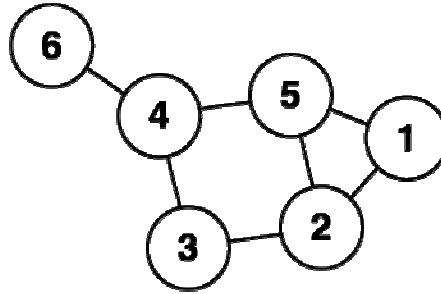


Figura 3.4 - Um grafo com 6 vértices e 7 arestas

Segundo Schwefel (1994) e West (2001), o primeiro teorema de teoria dos grafos foi proposto por Euler em 1736 para resolver o problema das pontes de Königsberg. Durante o século XIX problemas isolados sobre teoria dos grafos foram estudados, mas a partir do século XX houve um grande interesse nessa área, trabalhos importantes foram publicados como em kuratowski (1930) e Hakimi (1971). Exemplos de aplicações de grafos podem ser verificados em diversas áreas tais como: matemática, indústria eletrônica e de confecções, pesquisa operacional e redes de comunicação.

De acordo com Feofiloff *et al.* (2005), um grafo G pode ser definido como um par (V, A) em que V (vértices) é um conjunto não vazio de vértices e A (arestas) é um conjunto de pares ordenados de elementos de V . Em um grafo simples, dois vértices v e w são adjacentes se há uma aresta $a = (v, w)$ em G . Também recebem denominação de adjacentes duas arestas que compartilham o mesmo vértice. O conjunto formado por todos os vértices adjacentes a um vértice específico v recebe o nome de vizinhança de v e a cardinalidade desse conjunto é dita de grau v . Um grafo cujos vértices tenham o mesmo grau é considerado regular. A cardinalidade do conjunto V e A são chamadas de ordem e tamanho respectivamente. Além disso, se o grafo possuir ao menos um vértice com grau igual a zero, o grafo será denominado desconexo.

Cada vértice de V pode possuir um rótulo de identificação, neste caso classifica-se o grafo V como um grafo rotulado. Se cada aresta do sub-conjunto A estiver associada a um peso, o grafo será valorado. Um grafo $G(V,A)$ é classificado como: multigrafo, quando existem múltiplas arestas entre pares de vértices de G ; pseudografo, quando possui pelo menos uma aresta que começa e termina no mesmo vértice; e dígrafo, quando as arestas do grafo são orientadas. Há vários outros conceitos e discussões acerca da teoria dos grafos, mas os aspectos básicos apresentados serão suficientes para o entendimento da metaheurística de colônia de formigas utilizada neste trabalho.

4 SISTEMAS MULTIAGENTES

*“O binómio de Newton é tão belo como a Vénus de Milo.
O que há é pouca gente para dar por isso.”*

Álvaro de Campos⁸

Neste capítulo, alguns conceitos dentro da área de computação, importantes para o entendimento da modelagem do sistema computacional desenvolvido e que contextualizam o campo de sistemas multiagentes, são explanados, tais como: computação natural, inteligência artificial e inteligência artificial distribuída.

4.1 Computação natural

Segundo Castro e Zuben (2004), computação natural é a área da computação responsável pela produção de sistemas computacionais inspirados em sistemas naturais, incluindo sistemas biológicos, ecológicos e físicos. Dentre os seus objetivos têm-se:

- Desenvolver ferramentas matemáticas e computacionais para a solução de problemas complexos em diversas áreas do conhecimento;
- Projetar dispositivos que simulam, emulam, modelam e descrevem sistemas naturais;
- Utilizar mecanismos naturais, como cadeias de DNA e dispositivos mecânicos quânticos, como novos paradigmas de computação em substituição aos computadores atuais baseados em silício.

A computação natural compreende uma grande variedade de sub-áreas empregadas na otimização de problemas. Ressalta-se a computação inspirada na natureza que inclui todas as estratégias desenvolvidas a partir de ou inspiradas em algum mecanismo biológico ou natural. Como exemplos têm-se redes neurais, computação evolutiva, inteligência coletiva e os sistemas imunológicos.

Os trabalhos de Rechenberg (1973), Fogel (1966), e Holland (1975) deram origem à linha de pesquisa conhecida como computação evolutiva, composta pelos algoritmos evolutivos que usam idéias da biologia evolutiva para desenvolver algoritmos a serem empregados em tarefas de busca e otimização. Castro *et al.* (2004), apresenta uma

⁸ Álvaro de Campos (1890 - 1935) é um dos heterônimos mais conhecidos de Fernando Pessoa.

comparação entre os métodos evolucionários e algoritmo genético para problemas de confiabilidade de sistemas.

Segundo Bonabeau (1999) e Kennedy (2001), a inteligência coletiva (*swarm intelligence*) possui duas principais frentes de pesquisa: algoritmos baseados no comportamento coletivo de insetos sociais, e algoritmos baseados em comportamentos sócio-cognitivos humanos. No primeiro caso, a observação do comportamento coletivo de formigas e outros insetos levou ao desenvolvimento de algoritmos para a solução de problemas de otimização combinatória, agrupamento de dados, robótica coletiva, e outros. Algoritmos baseados em sócio-cognição são eficazes para a realização de buscas em espaços contínuos. Neste trabalho desenvolveram-se algoritmos baseados em colônia de formigas.

De acordo com Castro e Zuben (2004), a computação natural pode ser dividida em três grandes áreas: computação inspirada na natureza; computação com mecanismos naturais, e estudos sobre a natureza através da computação. Geralmente enfatizando modelos simplificados e abstratos da natureza, como ilustra a Figura 4.1. O tratamento simplificado ocorre visando tornar a computação tratável e destacar as características mínimas de um sistema que permite o uso de algum de seus aspectos particulares. Além disso, nem sempre são conhecidos os detalhes do sistema natural estudado.

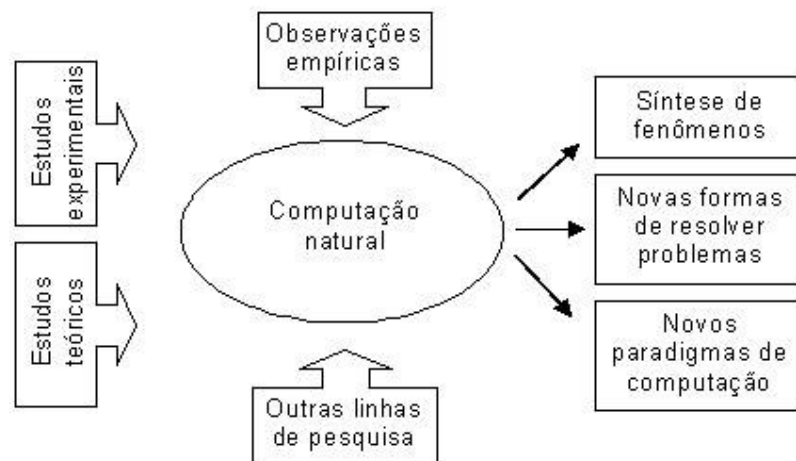


Figura 4.1 - Integração de linhas de pesquisa para o desenvolvimento da computação natural. Fonte: (CASTRO;ZUBEN, 2004)

Diversos trabalhos na área de computação natural vêm sendo desenvolvidos, como por exemplo, os apresentados em Fogel (2000), Dorigo e Stützle (2004) e Zomaya (2001), por fornecerem soluções alternativas, para problemas até então não resolvidos ou resolvidos de forma pouco eficiente.

4.2 Inteligência artificial

A Inteligência Artificial (IA) é uma área de pesquisa da ciência da computação que visa construir máquinas que funcionarão de forma autônoma em ambientes complexos e mutáveis. Seus estudos seguem como modelo o comportamento individual humano. A pesquisa nesse campo se iniciou após a segunda guerra mundial, mas fundamentou-se em áreas de conhecimento tais como: filosofia, matemática, economia, neurociência, psicologia, engenharia de computadores, teoria de controle e lingüística.

De acordo com Russel e Norvig (2004), o primeiro trabalho reconhecido como IA foi realizado por McCulloch e Pitts (1943) que propuseram um modelo de neurônios artificiais. Turing (1950) apresenta o teste de Turing, aprendizagem de máquina, algoritmos genéticos e aprendizagem por esforço. Após tentativas frustradas por parte da IA para o desenvolvimento de sistemas capazes de resolver uma grande quantidade de problemas (*General Problem Solvers – GPS*), os pesquisadores perceberam as limitações das abordagens utilizadas que propunham uma manipulação algorítmica de estruturas simbólicas. Havia uma visão procedural sugerindo que sistemas inteligentes poderiam ser projetados codificando conhecimentos especialistas em algoritmos específicos. Estes sistemas foram denominados de “sistemas especialistas”. Em meados de 1960, surgiram novos sistemas através da observação de fenômenos inteligentes naturais, ocasionando uma necessidade de dissociar redes neurais das outras técnicas clássicas de IA. Criou-se, então, uma nova linha de pesquisa denominada inteligência computacional que engloba redes neurais artificiais, computação evolutiva, sistemas nebulosos, inteligência coletiva e sistemas imunológicos artificiais.

Segundo Cohen (1995), em termos de metodologia, a IA adotou com firmeza o método científico. Para serem aceitas, as hipóteses devem ser submetidas a rigorosos experimentos empíricos, e os resultados devem ser analisados estatisticamente de acordo com sua importância.

4.3 Inteligência artificial distribuída

A Inteligência Artificial Distribuída (IAD) é uma das áreas da inteligência artificial que se preocupa principalmente com a criação de agentes inteligentes autônomos e com a comunicação de conhecimentos entre sistemas inteligentes. O modelo utilizado baseia-se no comportamento social, sendo a ênfase colocada em ações e interações entre agentes (ALVARES; SICHMAN, 1997). Segundo Bond e Gasser (1988), os principais problemas abordados em IAD são: descrição, decomposição e alocação de tarefas; interação, linguagem

e comunicação; coordenação, controle e comportamento coerente; conflito e incerteza; linguagens e ambientes de programação.

A IAD é dividida em duas áreas: resolução distribuída de problemas (RDP) e sistemas multiagentes (SMA). O sistema RDP soluciona um problema em particular sua abordagem consiste na decomposição do problema, resolução dos sub-problemas e síntese das soluções. O problema é resolvido por um conjunto de agentes, fisicamente distribuídos em diversas máquinas conectadas via rede. A motivação para sua implementação consiste, entre outros, na: velocidade da resolução do problema (distribuição de uma aplicação em várias máquinas); especialização inerente ao problema (monitoramento de uma grande área geográfica); necessidade dos resultados serem distribuídos (entrega distribuída).

4.4 Sistemas multiagentes

De acordo com Ribeiro (2001), Silva (2005) e Hübner (2003) a pesquisa sobre agentes de *software* se intensificou nos últimos 20 anos, tendo início na área de inteligência artificial distribuída (IAD) tornando-se tema de pesquisa da comunidade de Engenharia de *Software*. Os sistemas multiagentes (SMA) é uma sub-área da inteligência artificial distribuída que estuda a coletividade. Segundo Klügl (2002), SMA é adequado para modelar colônia de formigas, devido à facilidade de representação dos indivíduos ativos originais como agentes do modelo. Além disso, apresentam vocação para modelar sistemas complexos adaptativos como ressalta Hübner e Sichman (2003), pois possibilitam simular sistemas complexos, onde o conhecimento está distribuído, o controle não é passível de centralização e os recursos estão dispersos.

Considerando o desenvolvimento de sistemas, os SMA objetivam definir modelos genéricos de agentes, além de determinar interações e organizações que possam ser instanciadas dinamicamente em um dado problema. O paradigma dos sistemas multiagentes é motivado pela observação de alguns sistemas naturais, nos quais se percebe o surgimento de um comportamento inteligente a partir da interação de seus elementos. SMA estuda o comportamento organizado de agentes autônomos que cooperam na resolução de problemas que estão além das suas capacidades individuais de resolução. Evidenciam-se dois elementos para a compreensão de sistemas multiagentes: agente e ambiente.

Segundo Briot e Demazeau (2002), um agente é uma entidade lógica ou física capaz de cumprir uma missão que lhe é atribuída de maneira autônoma e em coordenação com outros agentes. Dado um determinado sistema, o *agente* seria cada uma de suas entidades ativas. O

conjunto de agentes forma uma *sociedade*. As entidades passivas podem ser designadas pelo termo *ambiente*. Alvares e Sichman (1997) definem *interação* como as trocas de informações que podem ocorrer entre os agentes. Tais trocas podem ser realizadas de modo direto ou de modo indireto. Agentes, assim como objetos, fornecem um conjunto específico de serviços para seus usuários. Um agente de *software* é dirigido por objetivos, conhecimento e um número de propriedades comportamentais tais como autonomia, adaptação, interação, colaboração, aprendizado e mobilidade.

Um sistema multiagente passa por duas etapas: concepção e resolução. Na concepção definem-se os modelos de propósito gerais para os agentes, para suas interações e para suas formas de organização. Já na resolução, um grupo de agentes adota estes modelos para resolver os problemas que lhe são apresentados. A Figura 4.2 ilustra esse processo. Os agentes podem ser reativos ou cognitivos.

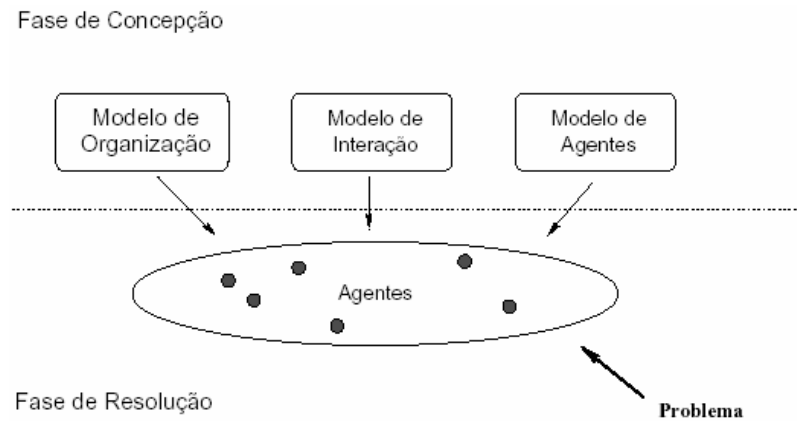


Figura 4.2 - Desenvolvimento ideal de sistemas multiagentes. Fonte: (HÜBNER, 2003)

4.4.1 Sistemas multiagentes reativos

Os agentes reativos escolhem suas ações baseados nas suas percepções sobre o ambiente. Inicialmente proposto por Brooks (1986), o agente apresenta as seguintes propriedades: normalmente possui representação de conhecimento implícita no código; por não possuir memória, não tem história dos fatos que aconteceram e das ações que executou; não tem controle deliberativo de suas ações; e as sociedades são formadas por muitos agentes.

Nestas sociedades, o interesse está voltado para a emergência de um comportamento global a partir da interação de um grupo grande de agentes. A arquitetura proposta por Russel e Norvig (1995) para o modelo reativo de agente é ilustrada Figura 4.3.

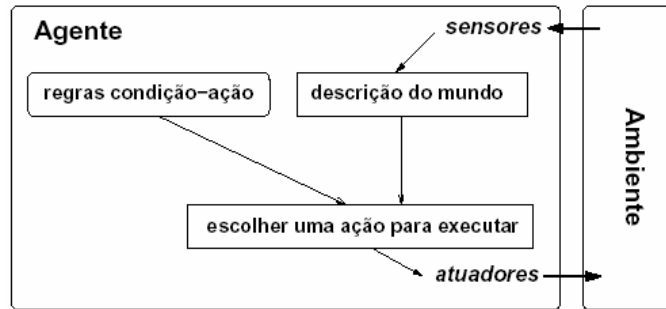


Figura 4.3 - Agente reativo. Fonte: (HÜBNER, 2003)

Nesta abordagem não é necessário sistemas complexos para resolver problemas complexos. Um exemplo é uma colônia de formigas: cada formiga isoladamente é uma entidade bem simples, entretanto, o trabalho realizado por uma colônia de formiga é bem complexo. Os agentes reativos são muito simples, as informações relativas ao seu comportamento estão no ambiente e suas reações dependem de sua percepção deste ambiente. Apesar de sua simplicidade o trabalho em grupo permite tarefas que não seriam possíveis individualmente.

4.4.2 Sistemas multiagentes cognitivos

Os agentes cognitivos raciocinam para construir um plano de ações que leva a um objetivo pretendido. Apresentam características particulares que os diferenciam de programas convencionais e dos agentes reativos, como se verifica na Figura 4.4.

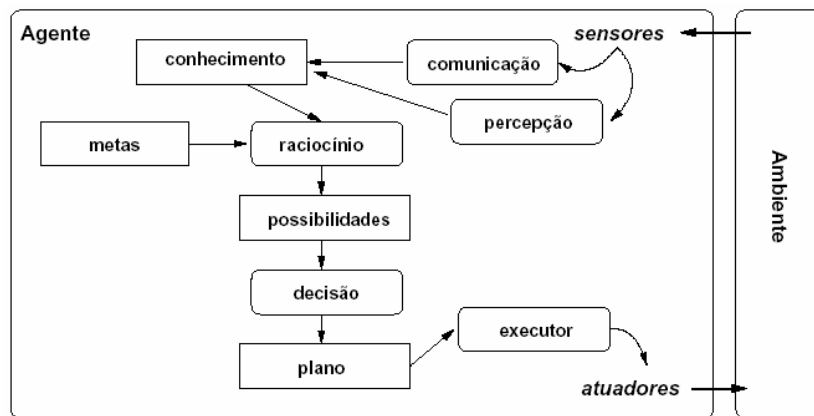


Figura 4.4 - Agente cognitivo. Fonte: (HÜBNER, 2003)

De acordo com Feber e Grasser (1991), os SMAs cognitivos são baseados em modelos organizacionais humanos, como grupos hierarquias e mercados. Os agentes cognitivos mantêm uma representação explícita de seu ambiente e dos outros agentes da sociedade,

podendo manter um histórico das interações e ações passadas. A comunicação entre os agentes é feita de modo direto, através do envio e recebimento de mensagens, além disso, seu mecanismo de controle é deliberativo, ou seja, tais agentes raciocinam e decidem sobre quais objetivos devem alcançar, que planos seguir e quais ações devem ser escutadas num determinado momento. Seu modelo organizacional é baseado em modelos sociológicos.

Há diversas classes de agentes cognitivos de acordo com o aumento da complexidade, tais como agentes: organizados, mantém perspectivas múltiplas sobre um determinado problema, obedecendo a leis e regras sociais; negociantes, contemplam um processo de resolução de conflitos por negociação; intencionais, representam internamente noções como intenções, engajamentos e planos parciais; e cooperativos, contém representações mútuas uns dos outros.

Um sistema multiagentes de agentes reativos é mais adequado na modelagem de formigas artificiais, pois neste modelo não é necessário que cada agente seja individualmente inteligente para alcançar um comportamento global inteligente. Além disso, os agentes são sensíveis ao ambiente e mudam seu comportamento se houver mudanças no mesmo.

5 COLÔNIA DE FORMIGAS

“O todo é mais do que a soma de suas partes.”

Aristóteles, Metaphysica

O presente capítulo aborda conceitos referentes à colônia de formigas, como as formigas se comportam e o modelo combinatório de otimização de colônia de formigas.

5.1 Definições gerais

Otimização com colônia de formigas (*Ant Colony Optimization* - ACO) é uma classe de algoritmos, inicialmente denominado *Ant System* proposto por Colormi (1991). Inspirado em formigas reais, é um algoritmo de busca paralelo baseado nos dados locais do problema e em uma estrutura de memória dinâmica que contém a informação do resultado previamente obtido. O comportamento coletivo que emerge da interação das diferentes linhas de busca mostrou-se eficaz em resolver problemas de otimização combinatória. Haken (1977), Nicolis e Prigogine (1977) afirmam que o comportamento de insetos sociais, em especial de formigas, tem sido muito estudado, devido à capacidade destes insetos realizarem tarefas complexas a partir de interações entre indivíduos simples.

Os insetos sociais conseguem atingir coletivamente padrões complexos de comportamento devido à combinação de dois processos: (i) auto-organização e (ii) *estigmergia*. O primeiro deles é um conceito que foi inicialmente introduzido nas áreas de Física e Química para descrever como processos microscópicos originavam estruturas macroscópicas em sistemas fora de seu estado de equilíbrio. Em sistemas biológicos auto-organizados a informação é adquirida e processada localmente mediante a interação de cada indivíduo com os vizinhos mais próximos, esta interação pode ser direta ou indireta. O processo de comunicação indireta foi denominado *estigmergia* (GRASSÉ, 1959).

Palazzo (1999) e Anderson (2002) identificam algumas propriedades que devem estar presentes em modelos de sistemas auto-organizados: possuir um grande número de componentes; possibilitar múltiplas interações locais entre os componentes; possuir a capacidade de realizar retro-alimentação; e apresentar características estocásticas. Segundo Anderson (2002), decorre como resultado de tais propriedades o chamado *fenômeno de emergência*, aspecto marcante de sistemas auto-organizados.

Segundo Klügl (2002), quando entidades que interagem localmente produzem um padrão ou comportamento observável numa escala global que não é diretamente dedutível a partir do comportamento local têm-se o fenômeno de emergência. Ou seja, a partir da interação de indivíduos simples que podem apresentar comportamento local aparentemente desordenado, tem-se comportamento global inteligente. A emergência pode ser verificada em vários contextos. Em uma cidade que cresce de forma desordenada, por exemplo, haverá com o tempo o agrupamento de pessoas do mesmo nível social em bairros próximos, mesmo sem planejamento ou acordo prévio.

Os sistemas complexos adaptativos são aqueles cujos componentes podem ser vistos como agentes capazes de se adaptar ao meio em que se encontram, alterando suas funções internas de processamento de informações. Caracterizam-se pela ausência de líder, por sua capacidade de realizar tarefas complexas que excedem a capacidade individual de seus membros. Como a ausência de um componente não interfere no desempenho do sistema, os sistemas complexos adaptativos não são criticamente dependentes de seus componentes. Observados do ponto de vista de seus componentes, tais sistemas são desbalanceados. No entanto, de uma perspectiva global, o sistema parece ser estável e ordenado. Essa característica denomina-se meta-balanceamento. Como observado por Klügl (2002), a capacidade de colônias de formigas realizarem, mesmo sem uma coordenação explícita, tarefas complexas que excedem a capacidade individual de seus membros trata-se de um fenômeno típico de emergência, muito comum em sistemas complexos adaptativos.

5.2 Comportamento das formigas

Características importantes do comportamento das formigas que podem ser modeladas para otimização de sistemas e serão abordadas neste tópico: comunicação química; teoria de busca de alimento; estratégias territoriais; aprendizado; orientação e memória.

Hölldobler e Wilson (1990) apontam que o sucesso das colônias de formigas torna-se possível graças à divisão de trabalho existente. A colônia está dividida em castas⁹ de trabalho, compreendendo as rainhas, os machos e as operárias. Enquanto o papel da rainha é exclusivamente pôr os ovos que darão origem aos novos indivíduos da sociedade, as operárias dividem entre si muitas tarefas segundo a sua idade e tamanho. Para as trabalhadoras mais novas e menores, por exemplo, ficam atribuídas as atividades de alimentação da rainha e

⁹ Em entomologia (ciência que estuda os insetos), o termo casta descreve os tipos de indivíduos morfológicamente diferentes e com tarefas especializadas que se encontram nas sociedades das formigas, abelhas e térmitas.

larvas e de limpeza do formigueiro. Já para as integrantes mais velhas, sobram os afazeres arriscados, como a proteção do ninho – para as formigas maiores – e a busca por novas fontes externas de alimento. Relativamente aos machos, estes parecem viver apenas o suficiente para fecundar as fêmeas.

A comunicação entre operárias é importante quando uma dada atividade da colônia está sendo realizada em grupo. Há diversas razões para haver comunicação entre as formigas, entre elas destaca-se o recrutamento de indivíduos para esgotar uma nova fonte de alimento. Esta propriedade está relacionada à capacidade das formigas de resolver problemas de otimização. A principal forma de comunicação ocorre através de substâncias químicas denominadas feromônios. As formigas percebem com as antenas o vapor criado pela difusão do feromônio no ar. Além disso, podem distinguir trilhas diferentes pelas variações da quantidade de moléculas presentes (percebendo gradientes de concentração molecular). Ao processo de orientação baseado na capacidade de diferenciar trilhas diversas segundo a sua concentração molecular de feromônio dá-se o nome de *osmotropotaxis*.

As dinâmicas de recrutamento são descritas através da interação entre uma realimentação positiva, o reforço da trilha pelas formigas recrutadas, e uma realimentação negativa, representada pela evaporação do feromônio e pelo decréscimo da frequência de construção da trilha, devido à exaustão desta substância (CAMAZINE *et al.*, 2001). Segundo Vittori (2005), o tipo de recrutamento utilizado por cada espécie de formiga constitui uma adaptação da mesma às condições do ambiente em que se encontra. O recrutamento em massa constitui a forma de comunicação mais avançada e mais utilizada pelas formigas na captura de alimento. O recrutamento e a formação das trilhas de feromônio são influenciados por diversos parâmetros, como (i) o tamanho da colônia; (ii) o tipo de alimento; (iii) a abundância de fontes de alimento e (iv) a presença de competidores.

Segundo Wilson (1971), o movimento de formigas em busca de alimento é probabilístico e baseado na quantidade de feromônio presente nas diversas trilhas a partir do ninho, as quais podem conduzir a uma mesma fonte ou a fontes diferentes de alimento. Quando vários caminhos conduzem a uma dada fonte, a formiga que utiliza o menor caminho consegue obter alimento e retornar ao ninho em um período de tempo menor que as formigas que utilizam caminhos maiores, tornando a concentração de feromônio nesta rota superior aos demais caminhos disponíveis. Assim também, elas apresentam a capacidade de variar a quantidade de feromônio depositada de acordo com o tipo, a quantidade e a qualidade do alimento descoberto. Algumas regras utilizadas pelas formigas são:

- Continue buscando um certo tipo de alimento se ele é aceito pelas companheiras do ninho, pois ele provavelmente irá satisfazer as necessidades da colônia; e
- Siga uma trilha se a quantidade de feromônio presente nesta é suficiente, pois ela irá provavelmente conduzir ao alimento.

Cada uma destas regras é executada individualmente de forma probabilística, com precisão limitada. Porém, quando envolvem um grande número de operárias, o padrão de comportamento emergente é mais complexo e preciso que individualmente (VITTORI, 2005).

De acordo com Vittori (2005), há diversas estratégias territoriais dependendo da espécie de formiga analisada. Os ninhos podem ser descentralizados para facilitar a patrulha, exploração e defesa de uma extensa área habitada, além de reduzir custos de transportes. O padrão de divisão do ambiente depende fortemente das estratégias de busca utilizadas por cada espécie em particular. Um modelo bastante estudado é o da trilha de feromônios: os indivíduos percorrem inicialmente trilhas de feromônio produzidas para o recrutamento de companheiras para novas fontes de alimento descobertas e depois se dispersam sobre a área de busca de forma individual. Após a captura do alimento, eles retornam ao ninho percorrendo as trilhas produzidas anteriormente, que persistem no solo por longos períodos de tempo. Como os recursos encontrados são praticamente estáveis, as formigas utilizam continuamente as trilhas previamente construídas.

Formigas da espécie *Messor pergandei* empregam uma estratégia de busca um pouco diferente das demais espécies, pois consideram a quantidade de alimento da fonte, elas exploram sucessivamente os diversos setores da área de busca existentes ao redor do ninho através de mudanças rotacionais de direção. A direção do movimento é modificada ou uma nova rota é estabelecida quando o suprimento de alimento da área de busca inicialmente explorada decresce.

Devido a sua capacidade de aprendizado, as formigas apresentam comportamento individual flexível. Ressalta-se duas formas de aprendizado empregadas pelas formigas: a habituação, que consiste na diminuição da intensidade de resposta a um acontecimento como resultado da experiência; e o associativo, que constitui a maioria dos casos de aprendizado demonstrados (HÖLLDOBLER; WILSON, 1990). Nesta categoria, os indivíduos apresentam respostas condicionadas pela associação de recompensas, como a presença de alimento, a estímulos previamente sem significado.

Segundo Hölldobler e Wilson (1990), para se utilizar os modelos de otimização na teoria de busca de alimento, a função objetivo seria a energia da colônia, a qual deve ser

maximizada, e as restrições do modelo seriam as limitações dos insetos que podem ser obtidas pelo conhecimento detalhado de mecanismos comportamentais e fisiológicos, como pistas para detectar as presas, nutrientes requeridos pela espécie e sua capacidade de memorizar experiências passadas. As formigas raramente são capazes de maximizar a energia da colônia através do cálculo de custos, benefícios e riscos de mortalidade de cada nova situação vivenciada. Ao invés disto, elas aplicam algumas regras, que constituem decisões rápidas baseadas em estímulos locais. A utilização individual destas regras permite a adequação dos membros da colônia à maioria das situações enfrentadas. Vittori (2005) apresenta diversos estudos que revelam que as formigas possuem as capacidades de orientação e memória, e as utilizam em conjunto com as trilhas de feromônio para retornar ao ninho pelo caminho mais curto, após terem descoberto uma fonte de alimento, ou para manter a exploração de uma fonte específica.

Uma interessante característica das formigas passível de modelagem para otimização de sistemas são as diversas formas de busca de alimento. Essa variedade deve-se, entre outros fatores à inexistência de diversas espécies que, de acordo com seu grau de evolução, agem no ambiente de forma diferente.

5.3 ACO Combinatório

Segundo Dorigo e Blum (2005), ACO é inspirado em formigas reais quando buscam por alimento. As formigas inicialmente exploram a área ao redor do ninho de forma aleatória. Quando uma formiga encontra alimento, ela o avalia e leva certa quantidade deste ao ninho. Durante a viagem de retorno, a formiga deposita uma trilha de feromônio no chão. A quantidade de feromônio depositada depende da quantidade e qualidade do alimento e irá guiar outras formigas até a fonte de alimento. Estas características de uma colônia de formigas reais são exploradas para resolver problemas de otimização combinatória.

O componente central do ACO algoritmo é o modelo feromonal, que é usado probabilisticamente no espaço de busca. Como mostra Blum e Sampels (2004), pode ser derivado para o modelo de um problema combinatório definido da seguinte forma:

Definição: Um modelo $P = (S, \Omega, f)$ de um COP¹⁰ consiste de:

- Um espaço de busca S definido sobre um conjunto finito de variáveis de decisão discretas e um conjunto Ω de restrições entre essas variáveis;
- Uma função objetivo $f : S \rightarrow \mathfrak{R}_0^+$ para ser minimizada;

¹⁰ Problema de otimização combinatória são abordados no capítulo 4, seção 4.3.

O espaço de busca S pode ser definido como um conjunto de variáveis discretas X_i , $i=1,\dots,n$, com valores $v_i^j \in D_i = \{v_i^1, \dots, v_i^{|D_i|}\}$. Uma variável é instanciada quando se atribui um valor v_i^j para a variável X_i que é denotado por $X_i \leftarrow v_i^j$. Uma solução $s \in S$ que satisfaz todas as restrições do conjunto Ω é uma solução factível de um dado COP. Se o conjunto Ω é vazio, P é chamado um modelo de problema irrestrito. Uma solução $s^* \in S$ é chamada um ótimo global se e somente se: $f(s^*) \leq f(s) \quad \forall s \in S$. O conjunto de solução de todos os ótimos globais é denotado por $S^* \subseteq S$.

Segundo Socha e Dorigo (2006), o modelo de um COP é usado para derivar o modelo do feromônio usado pelo ACO. Primeiro, uma instância de uma variável de decisão $X_i \leftarrow v_i^j$, é chamada componente da solução e denotada por c_{ij} . O conjunto de todos os possíveis componentes de solução é chamado de \mathbf{C} . Um parâmetro de trilha de feromônio T_{ij} é associado a cada componente c_{ij} . O conjunto de todos os parâmetros da trilha de feromônio T_{ij} é chamado de \mathbf{T} e seu valor é denotado por τ_{ij} . Este valor é usado para atualizar o algoritmo ACO durante a busca. Devido aos bons resultados obtidos através da utilização de ACO, vem crescendo o número de trabalhos recentes que o aplicam. Pode-se citar como exemplo (SOCHA *et al.*, 2003), (BLUM; DORIGO, 2004), (PELLERINI; DORIGO, 2007) e (SOCHA; DORIGO, 2007).

6 METODOLOGIA PROPOSTA E IMPLEMENTAÇÃO

“É senso comum pegar um método e experimentá-lo. Se falhar, admite francamente e tente outro. Mas acima de tudo, tente alguma coisa”.

Franklin Delano Roosevelt.

Dorigo e Stützle (2004) desenvolveram uma *metaheurística* baseada em colônias de formigas que abrange uma representação genérica do problema através de um grafo de construção de soluções e uma definição do comportamento da formiga. Obtiveram-se bons resultados para problemas combinatórios *NP-difícil*. Além disso, a modificação do algoritmo para problemas contínuos também alcançou resultados satisfatórios, como se verifica em Socha e Dorigo (2006). A Figura 6.1 apresenta de forma ilustrativa um grafo sendo percorrido por uma formiga.

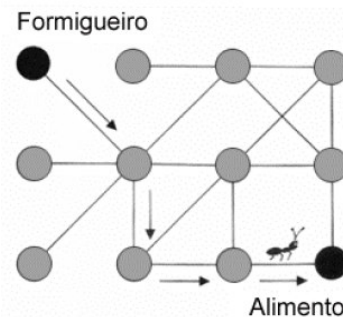


Figura 6.1 – Diagrama de um agente formiga se movimentando em um grafo (DORIGO; STÜTZLE, 2004)

Neste capítulo serão descritas a modelagem do sistema multiagente e sua implementação. As fórmulas apresentadas neste capítulo são adaptações de Dorigo e Stützle (2004) e Moreira e Sabóia (2006).

6.1 Metodologia proposta

Como apresentado no capítulo 3, um sistema multiagente de agentes reativos é mais adequado para modelar formigas artificiais, pois permite a representação da habilidade dos insetos reais em resolver conjuntamente problemas que muitas vezes excedem a capacidade de um único indivíduo. Logo, cada formiga será um agente autônomo do sistema que interage com os demais e com o ambiente. O ambiente será modelado como um grafo por onde as formigas podem transitar entre vértices.

O sistema multiagente para apresentar auto-organização deve: possuir um grande número de agentes; possibilitar múltiplas interações locais entre os agentes; possuir um mecanismo de retro-alimentação e possuir estocasticidade. Neste caso no sistema desenvolvido assegurou-se um ambiente com várias formigas que devem ser capaz de ler e escrever os pesos das arestas, que funcionarão como feromônio artificial, mediando à comunicação indireta entre tais agentes. O comportamento individual destes agentes em encontrar feromônio em uma aresta e depositar mais feromônio garante a retro-alimentação do sistema. Havendo um ambiente estocástico, o sistema será estocástico. Uma analogia entre os conceitos biológicos e sua representação no sistema pode ser visualizado na Tabela 6.1.

Tabela 6.1 – Analogias utilizados na modelagem do sistema

Conceito	Analogia
Formiga	Token capaz de percorrer o grafo, transitando entre os vértices através das arestas.
Ambiente de exploração	Grafo $G(V, A)$, onde V é o conjunto de vértices e A conjunto de arestas.
Feromônio	Peso p atribuído a cada uma das arestas do grafo.
Alimento	Solução factível.

A modelagem adotada para o ambiente do sistema multiagentes será a de um grafo. Obrigatoriamente, o grafo deve ser valorado, uma vez que a quantidade de feromônio depositado no ambiente é representada pelos pesos atribuídos às arestas. Na modelagem do ambiente ressalta-se a atualização da taxa de evaporação do feromônio. O feromônio é de fundamental importância no processo de troca de informações sobre os melhores caminhos até o alimento, pois funciona como um mecanismo de auxílio na exploração do grafo pelas formigas, evitando convergências prematuras. A fórmula 6.1 representa a taxa de evaporação.

$$\rho_{a_n} \leftarrow (1 - \rho)\rho_{a_n} \quad (6.1)$$

onde $\rho \in [0,1)$ é um parâmetro configurável e a_n são as arestas.

Os agentes correspondem às formigas artificiais. As formigas podem perceber os pesos (feromônios) dos vértices adjacentes, o seu estado atual, modificar o seu vértice-posição e seu estado interno. Além disso, podem atuar depositando feromônio.

De acordo com Dorigo e Stützle (2004), se a modelagem das formigas artificiais prevê o depósito de feromônio na ida e na volta do alimento, as formigas podem ficar presas em ciclos de feromônio, pois há probabilidade da formiga retornar aleatoriamente à sua própria marcação. Uma forma de evitar este ciclo seria através da adoção de um limiar de decisão ϕ . Neste trabalho, modelaram-se os estados das formigas da seguinte forma (ver Tabela 6.2).

Tabela 6.2 – Atuação do agente-formiga de deposição de feromônio

Estado	Exploradora	Experiente	Experiente Retornando
Ação	1	NADA	2

Quando a formiga k está configurada no estado “Exploradora” deposita um acréscimo feromonal Δ_{E_k} bem menor do que quando está no estado “Experiente Retornando” (Δ_{M_k}), ou seja $\Delta_{M_k} \gg \Delta_{E_k}$. O valor de Δ_{M_k} é inversamente proporcional ao comprimento do caminho que a formiga k tomou até o alimento (C_k). A fórmula 6.2 apresenta a atualização do feromônio nos dois casos.

$$\rho_{a_n} \leftarrow \rho_{a_n} + \Delta_{E_k}, \text{ para formiga } k \text{ no estado Exploradora} \quad (6.2)$$

$$\rho_{a_n} \leftarrow \rho_{a_n} + \Delta_{M_k}, \text{ para formiga } k \text{ no estado Experiente Retornando}$$

Tabela 6.3 - Atuação do agente-formiga de acordo com o valor do limiar

	Exploradora	Experiente	Experiente Retornando
$M \leq \phi$	1	2	3
$M > \phi$	2	2	3

O limiar de decisão pode ser visualizado na Tabela 6.3. Observa-se que se $M \leq \phi$, o valor de M , dado pelo maior valor das arestas adjacentes, é menor ou igual ao limiar de decisão. O ϕ obedece a seguinte fórmula:

$$\phi_k = \beta / C_k \quad (6.3)$$

Quando a atuação é configurada como 1, as formigas evitam probabilisticamente as arestas com maior feromônio, de modo que se dirijam às regiões do ambiente ainda pouco exploradas. Neste caso, a probabilidade $\bar{P}_{a_i}^k$ de uma formiga k escolher uma aresta a_i é:

$$\bar{P}_{a_i}^k = \begin{cases} \frac{\sum_{l \in N^k} p_{a_l} - p_{a_i}}{\sum_{l \in N^k} (\sum_{l \in N^k} p_{a_l} - p_{a_i})}, & \text{se } i \in N^k \\ 0, & \text{se } i \notin N^k \end{cases} \quad (6.4)$$

onde P_{a_n} é o peso da aresta a_n , e N^k é o conjunto de arestas adjacentes ao vértice v_n ocupado pela formiga, exceto a aresta utilizada anteriormente pelo agente para chegar ao vértice v_n .

Quando a atuação é configurada como 2, as formigas preferem probabilisticamente as arestas com valores mais altos de feromônio, emulando o fenômeno de *osmotropotaxis*. Neste caso, a probabilidade $\bar{P}_{a_i}^k$ de uma formiga k escolher uma aresta a_i é dada por:

$$\bar{P}_{a_i}^k = \begin{cases} \frac{P_{a_i}}{\sum_{l \in N^k} P_{a_l}}, se & i \in N^k \\ 0, se & i \notin N^k \end{cases} \quad (6.5)$$

onde P_{a_n} é o peso da aresta a_n , e N^k é o conjunto de arestas adjacentes ao vértice v_n ocupado pela formiga, exceto a aresta utilizada anteriormente pelo agente para chegar ao vértice v_n . O vértice atual da formiga ignora a posição de onde a formiga veio anteriormente. Esta atitude impede que formiga dê meia-volta no processo de caminhamento pelo grafo, exceto quando entra num beco sem saída, caso se posicione em um vértice com grau igual a um (1). Como o modelo é probabilístico, a escolha simula a ocorrência dos ruídos na percepção das formigas. Inicialmente, as arestas não devem possuir peso igual a zero (0), pois senão ter-se-á uma divisão indefinida.

Se a atuação for rotulada pelo valor 3, o agente-formiga retorna ao ninho de forma determinística, seguindo o caminho gravado em sua memória. O *agente-formiga* deve possuir capacidade de modificar o próprio estado interno a partir da análise do tipo do vértice em que se encontra, conjuntamente com o estado interno corrente.

Para otimizar o caminho do agente do alimento ao ninho, retirando possíveis ciclos dos caminhos armazenados na memória das formigas, Dorigo e Stützle (2004) apresentam um método de otimização. Antes de começar o processo de volta ao formigueiro segundo o caminho contido em sua memória, o agente deve tratá-lo, retirando os ciclos através do mecanismo ilustrado na Figura 6.2.

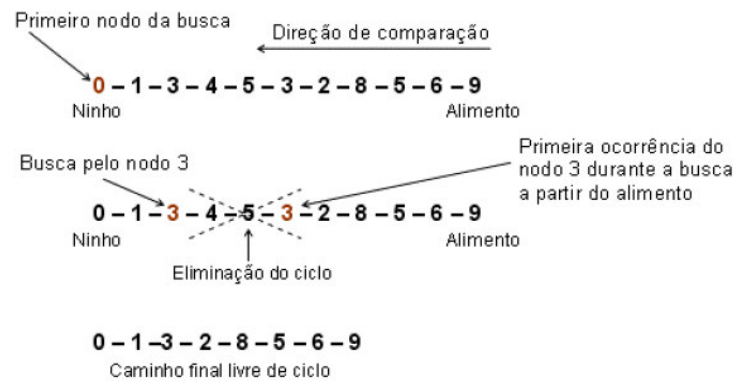


Figura 6.2 – Mecanismo de eliminação de ciclos (DORIGO; STÜTZLE, 2004)

Para a solução de problemas multiobjetivos, quando o *agente-formiga* encontra o alimento o caminho percorrido é armazenado e verifica-se a relação de dominância¹¹ entre a solução encontrada e as soluções já armazenadas, somente as soluções não dominadas são armazenadas para as próximas execuções.

6.2 Implementação

Segundo Moreira e Sabóia (2006), entende-se um agente (de um sistema multiagentes) como uma extensão de um objeto com características adicionais, como a autonomia, a interatividade e a adaptabilidade. Deste modo, tal informação sugere que o paradigma de programação orientado a objetos seja adequado para programar um sistema multiagentes. Sobre o termo orientação a objetos, Ricarte (2001) pressupõe uma organização de sistemas sobre uma coleção de objetos discretos incorporando estrutura e comportamento próprios. Esta abordagem de organização é essencialmente diferente do desenvolvimento tradicional de sistemas, onde estruturas de dados e rotinas são desenvolvidas de forma apenas fracamente acopladas. Devido à decisão pelo paradigma orientado a objetos e a necessidade de bom desempenho, a linguagem de programação utilizada foi a plataforma C++. Segundo Deitel e Deitel (2005), C++ é uma linguagem que evoluiu de C, desenvolvida por Bjarne Stroustrup no início dos anos 80 no *Bell Laboratories*. Apresenta várias características que melhoram a linguagem C, mas o mais importante é que fornece recursos para a programação orientada a objetos.

A linguagem de modelagem adotada foi UML (*Unified Modeling Language*) que permite visualizar os programas em diagramas padronizados. Para maiores informações sobre UML consultar Eriksson *et al.* (2004), Pender (2004) ou Guedes (2006). A modelagem foi desenvolvida no programa *StarUML*¹², projeto de código fonte aberto, com plataforma flexível e extensível que funciona na plataforma Win32.

As classes geradas foram organizadas em dois grandes pacotes: o pacote *graph*, que contém as classes que implementam o grafo, e o pacote *colony*, que contém a implementação das demais classes do programa, além da classe Simulator, como pode ser visto na Figura 6.3.

¹¹ Conceito de dominância abordado no capítulo 4

¹² Disponível gratuitamente na *internet* em <www.staruml.com>

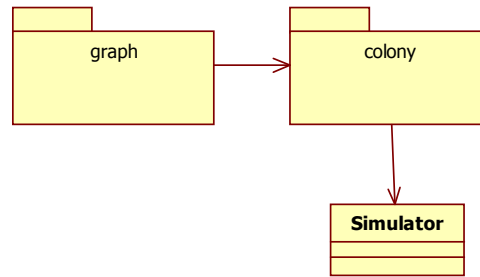


Figura 6.3 – Diagrama de pacotes

O pacote *graph* contém uma implementação orientada a objetos mais genérica para grafos, como se pode verificar no diagrama de classes apresentado na Figura 6.4.

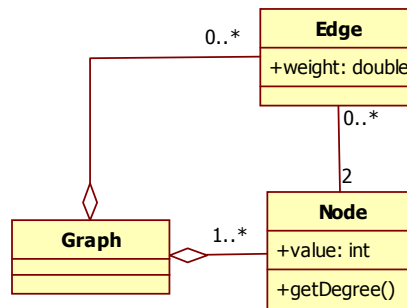


Figura 6.4 – Diagrama de classe do pacote graph

A classe *Graph* é composta pelas classes *Edge* e *Node*. Cada aresta conecta dois vértices e possui um valor de peso, como pode ser verificado no método construtor de *Edge* (Figura 6.5).

```

1  /** Metodo construtor parametrizado com nodes e peso */
2  Edge::Edge(int value, Node no1, Node no2, double weight){
3      this->no1 = no1;
4      this->no2 = no2;
5      this->weight = weight;
6  }
  
```

Figura 6.5 – Método construtor classe Edge

Na classe *Edge* encontram-se os métodos de configuração do feromônio. A classe *Node* implementa os vértices do grafo e pode ser configurada para possuir determinada quantidade de alimento, ou ser o ninho. O pacote *colony* contém a implementação das classes ambiente (*Environment*) e formigas (*Ant*). Na Figura 6.6 visualiza-se o diagrama de classe do ambiente do sistema com seus principais atributos e métodos.

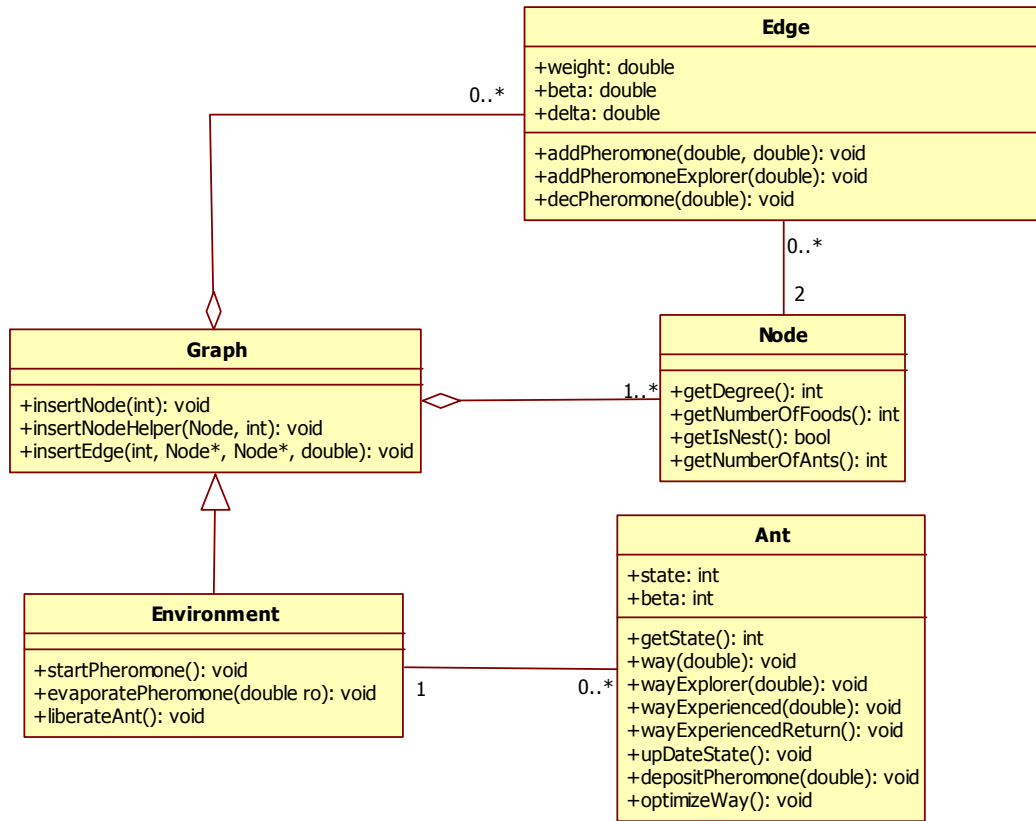


Figura 6.6 – Diagrama de classe do ambiente do sistema

A classe *Environment* herda da classe *Graph*, ou seja, todo ambiente é um grafo. Ressalta-se dois importantes métodos: *startPheromones()* e *evaporationPheromones(double)*. Através do primeiro método pode-se configurar a quantidade de feromônio inicial do ambiente; o segundo método simula a evaporação que ocorre naturalmente no ambiente. A classe *Ant* implementa completamente o *agente-formiga* modelado na seção 6.1. O objeto formiga possui um atributo *state* que expressa os possíveis estados da formiga (*stateExplorer*, *stateExperienced* e *stateExperiencedReturn*). Sobre o comportamento dos *agentes-formigas*, com respeito à capacidade de caminhar de um vértice a outro, o método *Ant::way(int)* apresentado na Figura 6.7, chama o método apropriado de acordo com o estado da formiga.

```

1  /* Implementa o caminho da formiga. O parâmetro: beta */
2  void Ant::way(int beta){
3      //Se o estado eh exploradora
4      if(this->state == stateExplorer){
5          this->wayExplorer(beta);
6      }
7      //Senão, se o estado eh experiente
8      else if(this->state == stateExperienced){
9          this->wayExperienced(beta);
10     }
11     //Senão retorna ao ninho
12     else{
13         this->wayExperiencedReturn();
14     }
15 }

```

Figura 6.7 – Método way da classe Ant

Em relação à capacidade dos *agentes-formigas* depositarem feromônio, verifica-se na Figura 6.8, implementada pelo método *depositPheromone(double)* parametrizado com o valor β , que o valor a ser depositado varia de acordo com o estado da formiga. Os objetos da classe *Simulator* ativam os objetos que implementam os agentes representantes das formigas e do formigueiro, um objeto da classe *Simulator* possui β e ρ , a quantidade de ciclos de execução e o ambiente como parâmetros.

```

1  /*Implementa o deposito de feromônio por parte da formiga. Parâmetro:beta*/
2  void Ant::depositPheromone(int beta){
3      //se o estado é exploradora
4      if(this->state == stateExplorer){
5          this->previousEdge->addPheromoneExplorer(beta);
6      }
7      //se o estado é experiente voltando e a ponte precedente não eh nula
8      else if ( (this->state == stateExperiencedReturn) &&
9              (this->previousEdge != NULL) ){
10         this->previousEdge->addPheromone(this->distPercurso, beta);
11     }
12 }

```

Figura 6.8 – Método depositPheromone da classe Ant

No próximo capítulo serão apresentadas soluções de problemas na área de engenharia de confiabilidade utilizando o sistema modelado no presente capítulo. Cada problema requer algumas alterações no sistema multiagente. As alterações, em geral, são aplicadas na classe *Environment*, bem como os métodos relacionados à análise de dominância das soluções. O método *verifyDominance(<double> solutionFitness, int nOfObjectives)*, recebe um vetor de soluções e compara-as entre si verificando quais delas não são dominadas e as armazenas no vetor *noDominatedSolution*, que posteriormente será analisado para gerar a curva de soluções não dominadas.

7 RESULTADOS

“Oh! Jogue fora a pior parte disso e deixe a parte mais pura com a outra metade.”

William Shakespeare.

No presente capítulo são apresentados problemas resolvidos através da metodologia discutida e implementada no capítulo 6. Inicialmente, valida-se o sistema multiagentes que foi denominado por “ACO-Multiagente” através da solução do problema da ponte dupla estendida e a comparação dos resultados obtidos com os presentes na literatura. Resolve-se através do mesmo sistema os problemas de alocação de redundância, neste caso comparando-se os resultados obtidos com os outros existentes na literatura, tais como Zhao *et al.* (2007); e problemas de manutenção preventiva, em que compare-se o modelo proposto com o trabalho de Samrout *et al.* (2005). Posteriormente, realiza-se a otimização da política de testes periódicos de um sistema auxiliar de água de alimentação de uma planta de potência nuclear.

7.1 Modelo da ponte dupla estendida

Proposto por Dorigo e Stützle (2004), este modelo de grafo impõe certa dificuldade às formigas em encontrar o melhor caminho. Como se pode observar, sempre que uma formiga parte do formigueiro, ela tem duas opções à sua frente: uma livre de ciclos, mas pior que o caminho ótimo, e a outra compreende os dois melhores caminhos, cujo tamanho é igual a cinco (5), bem como outros vários caminhos piores e cheios de ciclos. A Figura 7.1 mostra este modelo.

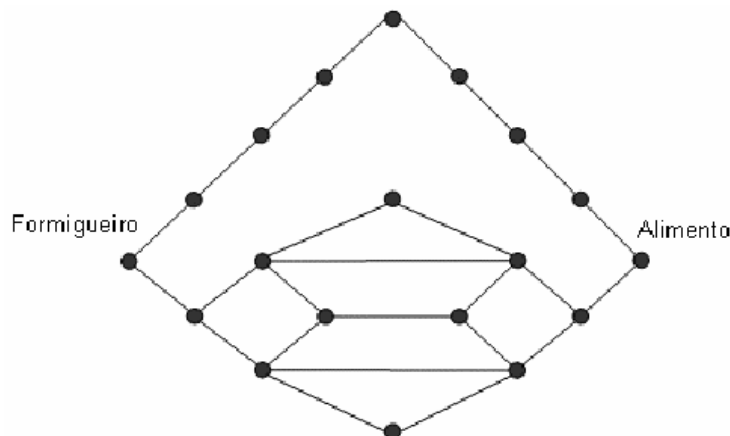


Figura 7.1 - Modelo da ponte dupla estendida. Fonte: (DORIGO; STÜTZLE, 2004)

O sistema desenvolvido neste trabalho alcançou bons resultados, como se pode verificar na Figura 7.2. Variou-se os parâmetros de configuração e os que apresentaram melhor desempenho foram $\beta = 25$ e $\rho = 0,02$, com o número de agente-formigas fixado em 100.

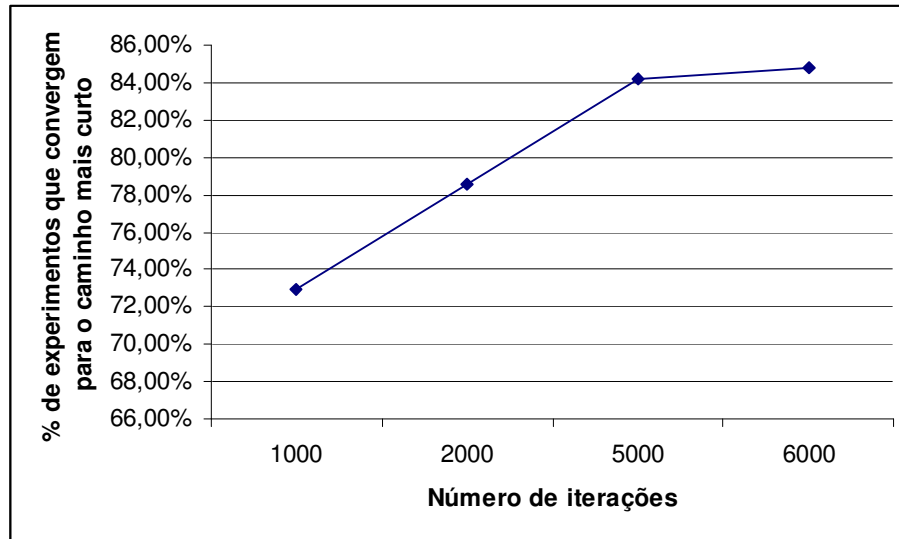


Figura 7.2- Desempenho do programa ACO-Multiagente

O sistema obteve rápida taxa de convergência inicial. Para mil (1000) execuções uma taxa de convergência de quase setenta e três por cento (73%), superior aos resultados obtidos por Moreira e Sabóia (2006) e Benzatti (2002), com cerca de três mil (3000) execuções o sistema já apresenta taxa de convergência superior a oitenta por cento (80%), porém acima de seis mil (6000) o ganho de convergência não é muito significativo em relação à quantidade de execuções necessárias. Comparando o resultado obtido com os apresentados por Moreira e Saboia (2006) com o sistema *Camponotus* e Benzatti (2002) com o sistema *AntSim* na Figura 7.3, verifica-se que a estabilização da porcentagem de convergência do sistema desenvolvido, do *Camponotus* e do *AntSim* foram bem próximos e que o sistema ACO-Multiagente alcança melhores resultados com menor número de interações.

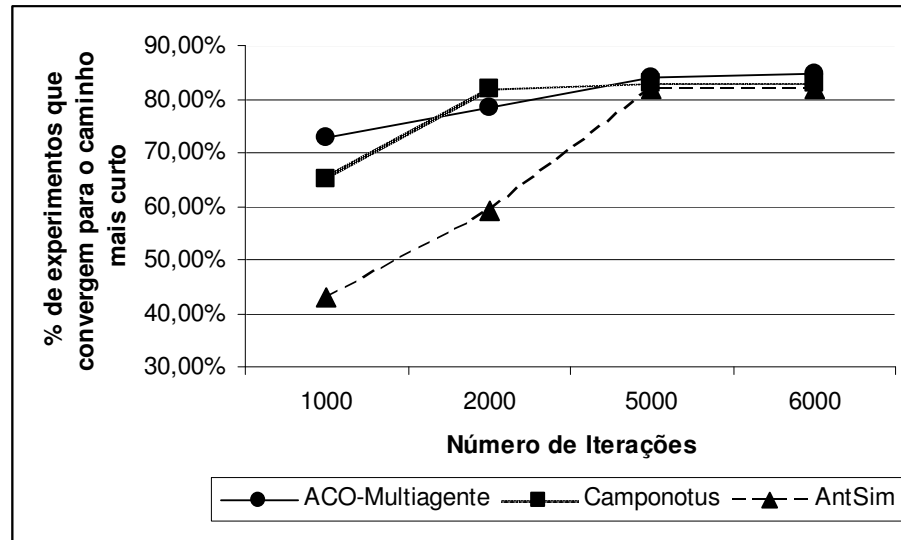


Figura 7.3 - Desempenho dos programas AntSim e Camponotus. Adaptado: (MOREIRA; SABOIA, 2006)

7.2 Problema de alocação de redundâncias

7.2.1 Formulação do problema

O problema considerado consiste em escolher os tipos e a quantidade de componentes para cada subsistema, de forma que o sistema como um todo tenha simultaneamente uma alta confiabilidade ou disponibilidade e um baixo custo. Assim, as variáveis de decisão do problema são as quantidades de cada tipo de componente que devem ser alocadas em cada subsistema. Logo essas variáveis de decisão são valores inteiros. A Figura 7.4 mostra a configuração de um sistema série-paralelo dividido em s subsistemas. Sendo que em cada subsistema i ao menos p_i componentes ativos são necessários e constituem a borda inferior do nível de redundância do subsistema i , a borda superior é PN .

A configuração do sistema pode ser representada por uma matriz $PN \times s$. O índice i da coluna representa o subsistema i , e o índice k da matriz representa a posição do componente no subsistema.

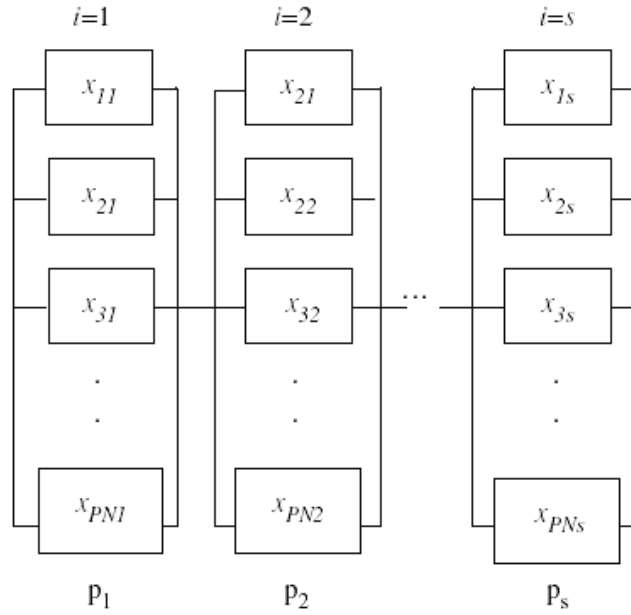


Figura 7.4 - Sistema série-paralelo. Fonte: (Zhao et al., 2007)

Define-se m_i como o tipo de componente para ser escolhido para o subsistema i e n_i como o número total de componentes redundantes no subsistema i . O tipo de componente em cada subsistema é ordenado de forma decrescente de acordo com a confiabilidade do conjunto de dados de entrada. O elemento x_{ki} ($x_{ki} = 1, 2, \dots, m_i, m_{i+1}$) representa o tipo de componente escolhido para a posição k do subsistema i . O índice m_{i+1} assinala que não foi alocado componentes redundantes nesta posição. Iniciando com o primeiro subsistema, a formiga seleciona o tipo de componente x_{ki} na posição k do subsistema i . Para cada componente há confiabilidade $R_i(x)$ e um custo $C_i(x)$ associados. Se n_i denota o número total de componentes redundantes no subsistema i , então:

$$n_i = \sum_{k=1}^{PN} x_{ik} / x_{ki} \quad x_{ik} \in (1, 2, \dots, m_i) \quad (7.1)$$

A confiabilidade de cada subsistema $R_i(x)$ é determinada por:

$$R_i(x) = 1 - \prod_{k=1}^{PN} (1 - r_{ix_{ki}}) \quad (7.2)$$

A confiabilidade do sistema pode ser determinada como:

$$R_s(x) = 1 - \prod_{k=1}^s R_i(x) \quad (7.3)$$

Da mesma forma, o custo do sistema é um somatório dos custos de cada subsistema,

$$C_S(x) = \sum_{i=1}^s C_i(x) = \sum_{i=1}^s \sum_{k=1}^{PN} C_{ix_{ki}} \quad (7.4)$$

O problema de alocação de redundância de um sistema série-paralelo pode ser formulado de acordo com Zhao *et al.* (2007) como:

$$\begin{aligned} \text{Maximizar: } R_S(x) &= \prod_{i=1}^s [1 - \prod_{k=1}^{PN} (1 - r_{ix_{ki}})] \\ \text{Minimizar: } C_S(x) &= \sum_{i=1}^s C_i(x) = \sum_{i=1}^s \sum_{k=1}^{PN} C_{ix_{ki}} \\ \text{Sujeito a: } & n_i \geq p_i \\ & \forall i, i = 1, 2, \dots, s \end{aligned} \quad (7.5)$$

Algumas considerações são requeridas para este tipo de problema. Como o objetivo não é mais encontrar o menor caminho e sim o menor custo e maior confiabilidade, o valor do feromônio a ser depositado pela formiga no estado Experiente¹³ deve levar em consideração a confiabilidade e o custo do nó. Logo, a partir da fórmula 6.2:

$$\rho_{a_n} \leftarrow \rho_{a_n} + \Delta_{M_k}, \text{ para formiga } k \text{ no estado Experiente Retornando} \quad (7.6)$$

Onde Δ_{M_k} leva em consideração a confiabilidade dividida pelo custo.

7.2.2 Exemplo de aplicação

O exemplo apresentado é um problema de alocação de redundância adaptado de Zhao *et al.* (2007) que consiste de um problema de otimização da confiabilidade de uma caixa de marcha com quatro estágios de transmissão. Cada estágio de transmissão é um subsistema com alguns pares de marcha e cada par de marcha é um tipo de componente com características iguais ou diferentes. O problema é decidir qual o tipo de par de marcha que deve ser escolhido e como maximizar a confiabilidade de cada estágio e minimizar o custo. Supondo que todos os pares de marcha são componentes ativos no estágio, a caixa de marcha é análoga a um sistema série-paralelo.

Considerando um número mínimo de pares de marcha $p_i=2$ e um número máximo $PN=5$ para todos os subsistemas, onde p_i é a quantidade de componentes ativos necessários que

¹³ A modelagem da formiga foi apresentada no capítulo 6.1

constituem a borda inferior do nível de redundância do subsistema i , e PN é a borda superior. Os dados de entrada dos componentes com seus respectivos custos pode ser visualizados na Tabela 7.1. Em que ri e ci representam respectivamente a confiabilidade e o custo para cada par de marcha em cada subsistema i , onde $i = 1, 2, 3, 4$.

Tabela 7.1 – Dados de entrada do problema de alocação de redundância

Pares de Marcha	Subsistema							
	1		2		3		4	
	r1	c1	r2	c2	r3	c3	r4	c4
1	0.855	3	0.743	5	0.828	9	0.74	10
2	0.706	5	0.882	6	0.842	7	0.922	10
3	0.931	5	0.874	2	0.779	7	0.855	15
4	0.737	7	0.783	7	0.911	7	0.864	13
5	0.805	6	0.9114	5	0.846	3	0.816	12

Para resolver esse problema utilizou-se o algoritmo multiobjetivo desenvolvido. Os tipos de componentes com seus respectivos custos e confiabilidades foram representados por Nós de um Grafo. A Tabela 7.2 apresenta os valores dos parâmetros utilizados no sistema.

Tabela 7.2 – Parâmetros utilizados no sistema multiobjetivo

Parâmetro	Valor
Quantidade de Formigas	60
Beta (β)	25
Taxa de evaporação (ρ)	0.03

Selecionou-se os 19 melhores resultados do sistema ACO-multiagentes, independente de serem soluções não dominadas, para poder efetuar comparação entre os sistemas ACSRAP¹⁴, desenvolvido por Zhao *et al.* (2007), e ACO-multiagentes, desenvolvido neste trabalho. Na Tabela 7.3 pode-se analisar as restrições e as simulações resultantes. Ressalta-se que C_{max} corresponde à restrição de custo máximo, C_s é o custo do sistema e $Max. R_s$ é a confiabilidade máxima obtida pelo sistema, no caso a denominação sistema refere-se aos sistemas analisados (ACSRAP ou ACO-Multiagente).

¹⁴ Sistema de colônia de formigas para problema de alocação de redundância (*Ant Colony System for Redundancy Apportionment Problem - ACSRAP*) desenvolvido por Zhao *et al.* (2007).

Tabela 7.3 – Resultados obtidos usando ACSRAP e ACO-Multiagentes

No.	Restrições Cmax	Sistema			
		ACS-RAP		ACO- Multiagente	
		Cs	Max. Rs	Cs	Max. Rs
1	40	40	0.9861	40	0.988
2	55	55	0.9973	52	0.9968
3	65	58	0.9977	55	0.9977
4	60	59	0.9968	56	0.9979
5	60	58	0.9977	55	0.9968
6	60	60	0.9985	58	0.9980
7	60	60	0.9987	59	0.9983
8	65	59	0.9968	62	0.9990
9	65	58	0.9977	60	0.9987
10	65	65	0.9988	64	0.9992
11	65	64	0.9990	60	0.9973
12	70	59	0.9968	58	0.9963
13	70	66	0.9988	65	0.9995
14	70	65	0.9990	59	0.9973
15	70	70	0.9992	66	0.9995
16	75	59	0.9968	65	0.9981
17	75	66	0.9988	69	0.9997
18	75	71	0.9992	70	0.9992
19	75	70	0.9995	70	0.9997

Para comparar a performance dos dois sistemas calculou-se a porcentagem da qualidade dos pontos obtidos através da divisão dos dados fornecidos na Tabela 7.3 (confiabilidade pelo custo). A partir do gráfico apresentado na Figura 7.5 pode-se analisar os resultados obtidos. Aplicou-se o teste não-paramétrico de Mann-Whitney e obteve-se um p-valor de 0.470, logo não há evidência para rejeitar a hipótese nula de igualdade das médias. Ou seja, os dois sistemas obtiveram desempenhos semelhantes.

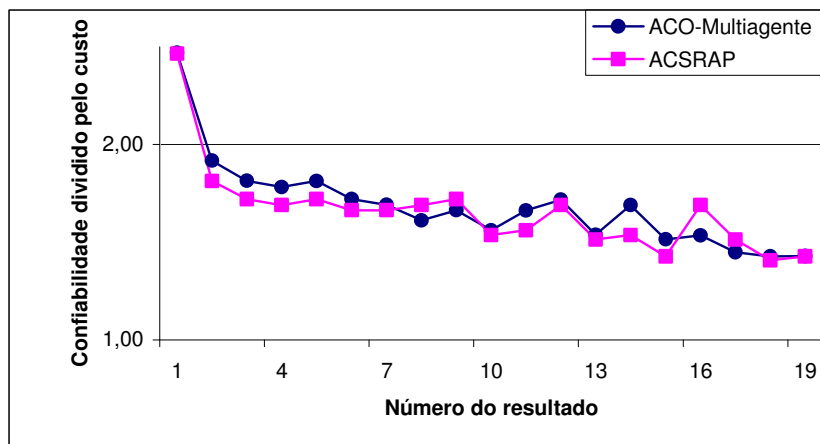


Figura 7.5 – Comparação entre ACO-Multiagente e ACSRAP

O sistema ACO-Multiagentes usou 28 interações com 60 formigas. Quando se avaliou a relação de dominância entre essas soluções, restaram 15 soluções não-dominadas que formam a fronteira de Pareto apresentada no gráfico da Figura 7.6. Percebe-se que a partir do custo de 68 unidades monetárias o aumento do custo não resulta em melhoria significativa da confiabilidade.

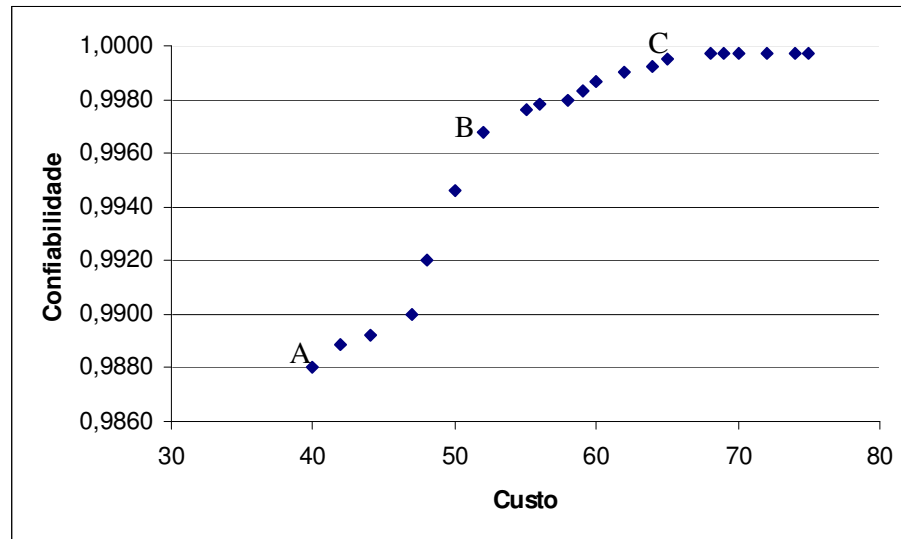


Figura 7.6 - Fronteira de soluções não-dominadas para o exemplo RAP¹⁵

A solução **A** apresenta um custo de 40 unidades monetárias para uma confiabilidade de 0.9880 (98,80%). Já a solução **B** apresenta um custo de 52 unidades monetárias para uma confiabilidade de 0.9968 (99,68%). Verifica-se que houve um aumento de custo de 12 unidades monetárias para melhoria de confiabilidade de 0,0088. Porém ao se comparar a solução **C** o custo é de 65 unidades monetárias para uma confiabilidade de 0.9995 (99,95%), comparando com a solução **B** houve um ganho de 0,0027 a um custo de 13 unidades monetárias.

7.3 Problema de otimização de uma política de testes periódicos

Nesta seção, inicialmente apresenta-se a resolução proposta para sistemas periodicamente inspecionados utilizando o sistema ACO-multiagente, seguido de um exemplo com um único objetivo retirado da literatura. Finalmente apresenta-se o problema do sistema auxiliar de água de alimentação e sua resolução através do sistema ACO-multiagentes.

¹⁵ Problema de alocação de redundância (*Redundancy Apportionment Problem – RAP*)

7.3.1 Resolução

O sistema é montado usando a mesma estrutura do problema de alocação de redundância. Cada componente tem uma lista de possíveis tempos de intervenção. Estes tempos são gerados aleatoriamente no intervalo [LI, LS] de acordo com uma distribuição uniforme, em que LI é o limite inferior e LS é o limite superior, como pode verificar na Figura 7.7.

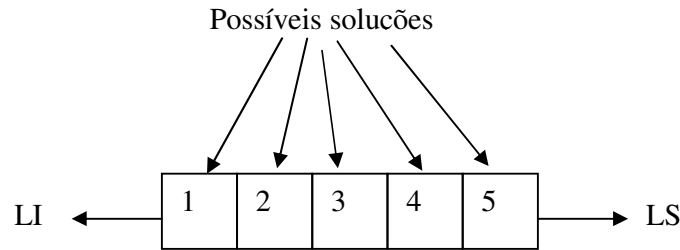


Figura 7.7 – Lista de componentes Adaptado: (Samrout, 2005).

Cada formiga k constrói uma solução para resolver este problema se movimentando pelo grafo. Durante uma interação t , cada formiga $k=1\dots m$ realiza um percurso $T_p^k(t)$. Este percurso contém possíveis tempos para fazer testes periódicos nos componentes do sistema. O número de elementos do vetor é o número de componentes para o qual se deve determinar as datas de intervenção. Cada percurso $T_p^k(t)$ é um vetor candidato para ser a melhor solução encontrada pelo método da Figura 7.8.

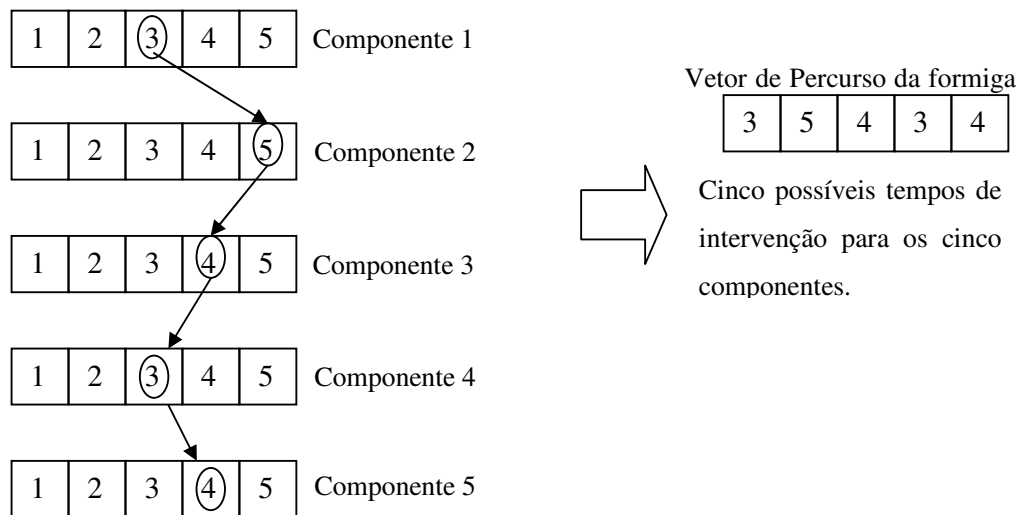


Figura 7.8 – Exemplo de construção de solução (1 formiga e 5 componentes). Adaptado: (Samrout, 2005)

7.3.2 Exemplo de validação

Este exemplo, adaptado de Samrout *et al.* (2005), consiste em encontrar um vetor T_p ótimo de soluções para um sistema de componentes periodicamente testados. Inicialmente deseja-se encontrar uma política de testes periódicos que minimize o custo do sistema, com respeito à disponibilidade, esta tem como restrição $A(t) \geq A_0$ para todo t , tal que $0 < t \leq T_M$, dado um tempo de missão T_M . Para este exemplo, assume-se que a política de testes periódicos melhora a confiabilidade dos componentes para “tão bom quanto novo”.

Bris *et al.* (2003) propuseram um método usando algoritmo genético para aproximar-se do vetor ótimo de testes periódicos. O método proposto é semelhante à fórmula 7.4 para o cálculo do custo do sistema de um problema de alocação de redundância, pois ambos os problemas tratam de um sistema série-paralelo. Porém para o cálculo do vetor de testes periódicos, acrescenta-se na formula $n_{e(i,k)}$, adaptado de Bris *et al.* (2003), tem-se:

$$C_{PM} = \sum_{k=1}^K \sum_{i=1}^{E_k} n_{e(i,k)} C(e(i,k)) \tag{7.7}$$

onde $C(e(i,k))$ é o custo constante de uma inspeção para o i -ésimo componente no k -ésimo subsistema paralelo, K é o número de subsistemas em paralelo, E_k é o número de componentes em série do subsistema K , $n_{e(i,k)}$ representa o número total de inspeções para o i th componente no k th subsistema. Numericamente é a parte inteira da fração:

$$n_{e(i,k)} = 1 + \left\lceil \frac{T_M(e(i,k)) - T_0(e(i,k))}{T_p(e(i,k))} \right\rceil \tag{7.8}$$

dado que T_M é o tempo de missão, T_0 é o primeiro tempo de inspeção e T_p o período de inspeção de um dado componente. A política de manutenção ótima é encontrada para um sistema série-paralelo cuja estrutura pode ser visualizada na Figura 7.9.

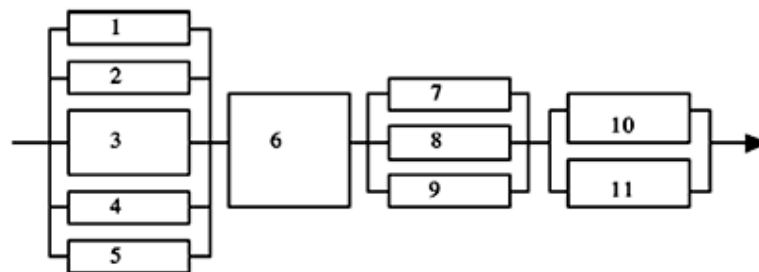


Figura 7.9 – Estrutura do sistema em paralelo-série. Fonte: (Bris *et al.*,2003)

O cálculo foi efetuado para tempo de missão de cinquenta anos com restrição de disponibilidade $A_0 = 0.8$. Uma comparação entre o método ACS de Samrout *et al.* (2005) pode ser encontrado na Tabela 7.4. Ressalta-se que C_s é o custo do sistema, no caso a denominação sistema refere-se aos sistemas analisados (ACS ou ACO-Multiagente).

Tabela 7.4 – Vetor T_p obtido com os melhores custos usando ACS e ACO-Multiagentes

N° componente	Período Inicial (T_0)	Sistema	
		ACS	ACO-Multiagente
		C_s	C_s
1	18	14.72	10.6
2	15	15.42	13.7
3	12	11.24	13.4
4	13	14.66	14.29
5	20	13.75	13.71
6	10	15.06	14.52
7	16	12.55	11.58
8	12	15.19	12.5
9	11	13.20	14.25
10	10	15.09	14.81
11	10	13.24	13.93

O teste confirmou o resultado apresentado no artigo Samrout *et al.* (2005), tanto para o sistema ACS quanto para o ACO-Multiagente o melhor custo obtido foi de 153.9.

7.3.3 Definição do sistema auxiliar de água de alimentação

Este exemplo é uma adaptação de Garcia (2006) e trata da otimização da política de testes periódicos para um sistema auxiliar de água de alimentação de uma típica planta de potência nuclear do tipo PWR de dois ciclos. Neste exemplo, busca-se a maximização da confiabilidade do sistema e a minimização do custo.

O sistema auxiliar de água de alimentação (“*Auxiliary Feed Water System*” - AFWS) é composto de dois subsistemas: um deles é composto por uma turbo bomba (TDP), que tem a capacidade de fornecer cem por cento das necessidades dos geradores de vapor (SG); o outro, é composto por duas bombas motorizadas (MDP), onde cada uma atende a cinquenta por cento das necessidades dos SG. Em outras palavras, o funcionamento da turbo bomba é suficiente para o bom funcionamento do AFWS ou o funcionamento das duas MDP. O sistema descrito pode ser interpretado como um sistema paralelo-série (Figura 7.10).

Em condições normais a água utilizada é proveniente do tanque auxiliar de água de alimentação (“*Auxiliary Feed Water Tank*” - AFWT). Numa planta típica do tipo PWR, o AFWS deve desempenhar as seguintes funções básicas:

- Suprir os geradores de vapor em caso de perda do sistema de água de alimentação principal;
- Manter o nível de água nos geradores de vapor de modo a remover o calor residual gerado pelo reator enquanto este se encontra a uma potência inferior a dez por cento da potência nominal.

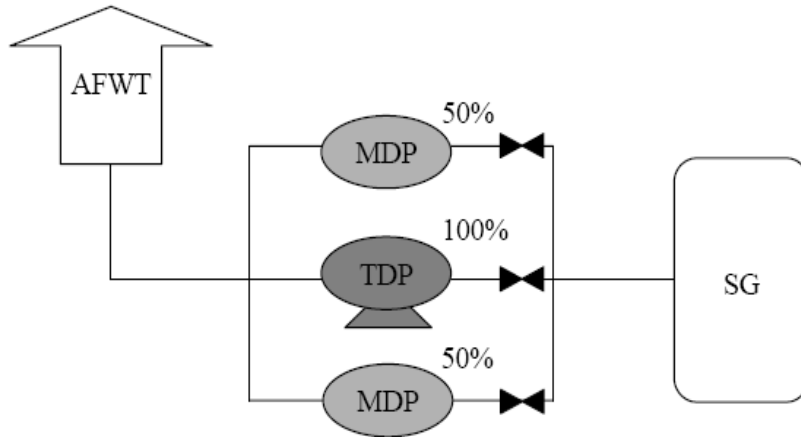


Figura 7.10 – Diagrama simplificado do AFWS

7.3.4 Cálculo do custo e da confiabilidade

Nesta seção, apresenta-se os modelos para o cálculo da confiabilidade de um componente, modelo este adaptado de Lapa *et al.* (2006), e o custo que foi calculado de acordo com a equação (7.7). O modelo para o cálculo da confiabilidade consiste de uma generalização do modelo proposto por Lewis (1996).

Considere $R(t)$ a confiabilidade de um componente que está suscetível a sofrer manutenção corretiva ou está sujeito a uma política de manutenção preventiva, mas ainda não sofreu intervenção de manutenção até o tempo t , onde t é o tempo de operação. Sendo $T_m(i)$ a data de calendário para a i -ésima intervenção de manutenção para o componente m , T_{mis} o tempo de missão e $T_m(ult)$, a data da ultima intervenção realizada até o tempo t .

$$R_m[t, T_m(i), T_m(ult)] = R[t - T_m(ult)] \prod_{i=1}^{ult} R[T_m(i) - T_m(i-1)], \quad T_m(ult) \leq t < T_{mis} \quad (7.9)$$

Considerando a influência da política de manutenção sofrida por um componente sobre a operação de todo o sistema, assume-se que o componente está fora de operação durante o período de tempo de manutenção $\Delta_m(i)$.

$$R_m[t, T_m(i), T_m(ult)] = \begin{cases} R[t - T_m(ult)] \prod_{i=1}^{ult} R[T_m(i) - T_m(i-1)] & T_m(ult) \leq t < T_{mis} \\ 0 & T_m(i) \leq t < T_{mis}(i) + \Delta_m(i) \end{cases} \quad (7.10)$$

A distribuição Weibull foi escolhida por ser uma boa representação para componentes que sofrem o efeito da idade (aumento da taxa de falha com o tempo). Típicos valores para taxa de falha e custos de manutenção foram usados de acordo com Harunuzzaman e Aldemir (1996). O objetivo é encontrar uma ótima política de testes para o sistema descrito durante um período de 480 dias. Os testes adotados e os tempos de interrupção de manutenção para cada componente do AFWS (Tabela 7.5) são similares aos publicados em Harunuzzaman e Aldemir (1996) considerando componentes similares.

Tabela 7.5 – Tempos de parada para os componentes do sistema AFWS (h)

Componente	Descrição	Tempo da Intervenção (h)	Tempo da Manutenção (h)
V1	Válvula 1	1	8
V2	Válvula 2	1	8
V3	Válvula 3	1	8
MDP1	Motor-driven pump 1	1	72
MDP2	Motor-driven pump 2	1	72
TDP	Turbo-driven pump	1	96

Para resolver esse problema utilizou-se o algoritmo multiobjetivo desenvolvido. Os tipos de componentes com seus respectivos custos e confiabilidades foram representados por Nós de um Grafo. A Tabela 7.2 apresenta os valores dos parâmetros utilizados no sistema. Segundo Pereira e Lapa (2003), baseado conhecimento especialista, considera-se um passo de tempo de 10 dias, suficiente para a aplicação prática.

Tabela 7.6 – Parâmetros utilizados para o problema AFWS

Parâmetro	Valor
Quantidade de Formigas	40
Beta (β)	20
Taxa de evaporação (ρ)	0.03

Os vetores com as melhores políticas de teste para cada componente podem ser visualizados na

Tabela 7.7.

Tabela 7.7 – Melhor política de testes encontrada

Componente	Testes Programados
V1	{47, 117, 137, 187, 277, 307, 357, 387, 417, 427, 447, 457, 467 }
V2	{46, 96, 146, 196, 256, 286, 316, 346, 366, 416, 436, 456, 466}
V3	{22, 82, 172, 232, 292, 312, 332, 352, 382, 402, 412, 422, 432, 442, 452, 462}
MDP1	{48, 108, 138, 188, 228, 258, 288, 318, 348, 368, 388, 408, 428, 448, 458}
MDP2	{32, 72, 112, 162, 202, 242, 262, 302, 332, 352, 382, 402, 422, 442, 462}
TDP	{83, 133, 153, 213, 263, 313, 353, 393, 423, 443, 463}

A Figura 7.11 apresenta os resultados obtidos pela maximização da confiabilidade e minimização dos custos, onde somente os valores para as funções objetivos que representam as soluções não dominadas são mostradas.

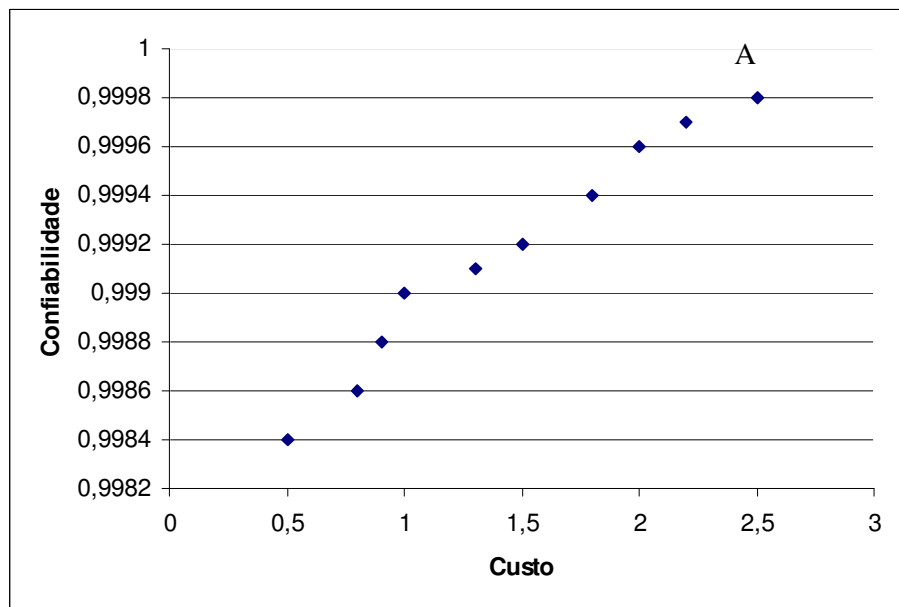


Figura 7.11 – Resultado da otimização multiobjetivo para o problema AFWS

A Tabela 7.7 apresenta a melhor política de testes encontrada (para cada componente), considerando-se a maior confiabilidade obtida 0.9998 (99,98%) a um custo de 2,5 unidades monetárias, que corresponde ao ponto A na Figura 7.11.

8 CONCLUSÕES E PERSPECTIVAS FUTURAS

“Fechei vossas palavras na memória; a chave vós mesmo a guardareis.”

William Shakespeare.

Neste capítulo são apresentadas às conclusões obtidas neste trabalho, e em seguida são feitas recomendações para trabalhos futuros de pesquisa.

8.1 Conclusões

Nesse trabalho desenvolveu-se um sistema multiobjetivo baseado em colônia de formigas para solução de problemas na área de engenharia de confiabilidade. Nos problemas abordados buscava-se soluções satisfatórias para os objetivos considerados – métricas de desempenho (confiabilidade) e custos associados. O algoritmo desenvolvido utiliza os conceitos de dominância e não-dominância, permitindo, dessa forma, o tratamento multiobjetivo dos problemas relacionados a projetos de sistemas e política de testes periódicos. A elaboração desse trabalho possibilitou um conhecimento mais aprofundado dos temas abordados: projetos de sistemas baseados em confiabilidade, otimização multiobjetivo, colônia de formigas, além da linguagem de programação C++, utilizada na implementação do sistema ACO-Multiagentes.

O sistema ACO-Multiagentes desenvolvido mostrou-se eficiente tanto para problemas de menor caminho, obtendo resultados para problemas com vários ótimos locais como foi o caso do exemplo da ponte dupla estendida no capítulo 7, quanto para problemas de alocação de redundância e testes periódicos. A velocidade de convergência também deve ser observada, demorando menos de 1 minuto para a solução de todos os problemas apresentados. Comparando-o a outros algoritmos baseados em colônia de formigas aplicados em problemas de engenharia de confiabilidade, o sistema ACO-Multiagentes encontrou a solução de forma mais rápida com um número menor de interações e formigas, atribui-se isso ao fato do modelo proposto considerar o retorno da formiga ao ninho, trabalhando com concentrações de feromônio.

A metodologia proposta e a correspondente implementação foram aplicadas para resolução de problemas de alocação de redundância e otimização de testes periódicos de aplicação nuclear, com a adição dos atributos listados abaixo:

- Facilidade de implementação;
- Resultados compatíveis (às vezes um pouco melhores) com os apresentados pela literatura, porém com menor esforço computacional.
- Obteve-se bons resultados para o problema de otimização da política de testes periódicos para um sistema auxiliar de água de alimentação de uma planta de potência nuclear do tipo PWR, em que se maximizou a confiabilidade e minimizou-se o custo, este tipo de problema multiobjetivo ainda não havia sido resolvido pela literatura através de colônia de formigas.

8.2 Limitações

A principal desvantagem dos métodos heurísticos de otimização é o fato de eles não garantirem o ótimo. Com respeito aos exemplos apresentados, tanto para o exemplo de alocação de redundância quanto para política de testes periódicos com um único objetivo, o resultado alcançado, apesar de ser obtido em menor tempo, não foi superior aos resultados já apresentados na literatura.

8.3 Perspectivas futuras

Tendo em vista a potencialidade do sistema desenvolvido pode-se aplicá-lo a um grande número de problemas reais multiobjetivos. Propõe-se verificar a viabilidade da melhoria do sistema através da utilização de outras técnicas junto como colônia de formigas como processo semi-markov visando à otimização da disponibilidade e do custo.

Propõe-se avaliar sistemas reparáveis sujeitos a reparos imperfeitos, por meio de processos de renovação generalizados, além de otimizar confiabilidade/disponibilidade e custo para outras configurações de sistemas, como a série-paralelo.

Propõe-se também comparar o desempenho do algoritmo desenvolvido com outras heurísticas tais como algoritmos genéticos multiobjetivo e otimização por enxame de partículas (PSO).

REFERÊNCIAS BIBLIOGRÁFICAS

- ALVARES, L. O. SICHMAN, J. S. Introdução aos Sistemas Multiagentes. In: *XVII CONGRESSO DA SBC*, Brasília, 1997.
- ANDERSON C. Self-organization in relation to several similar concepts: are the boundaries to self-organization indistinct? *Biol. Bull*, n. 202, p. 247- 255, 2002.
- ARISTÓTELES. *Metaphysica*. Oxford: Ed. W. Jaeger, 1960.
- ARROYO, J. E. *Heurísticas e metaheurísticas para otimização combinatória multiobjetivo*. Campinas, 2002. (Doutorado - UNICAMP).
- BENZATTI, D. *Emergent Intelligence*. 2002. Disponível em: <<http://aidepot.com/Essay/SocialInsects.html>>. Acesso em 20 de jul. 2007.
- BONABEAU E., DORIGO, M.; THERAULAZ, T. *Swarm intelligence: from natural to artificial systems*. New York, Oxford University Press, 1999.
- BOND, A.; GASSER L. *Readings in distributed artificial intelligence*. San Mateo, Morgan Kaufman, 1988.
- BLUM, C.; DORIGO, M. The hyper-cube framework for ant colony optimization. In: *IEEE Transactions*. vol.34, p. 1161-1172, 2004.
- BLUM, C., SAMPELS, M. An ant colony optimization algorithm for shop scheduling problems. In: *Journal of Mathematical Modelling and Algorithms*, pág. 285–308, 2004.
- BRIOT, J.-P.; DEMAZEAU, Y. *Principes et architecture des systemes multi-agents*. Paris: Hermes, 2002.
- BRIS, R.; CHATELET, E.; YALAOUI, F. New method to minimize the preventive maintenance cost of serie-parallel system. . In: *Reliability Engineering & System Safety*, vol. 82, p. 247–255, 2003.
- BROOKS, R. A robust layered control system for a mobile robot. In: *IEEE journal of robotics and automation*, v.2, n.1, p.14-23, março, 1986.
- BUSACCA, P. G.; MARSEGUERRA, M.; ZIO, E. Multiobjective optimization by genetic algorithms: application to safety systems. In: *Reliability Engineering & System Safety*, vol. 72, p. 59–74, 2001.

- CAMAZINE, S.; DENEUBOURG, J.-L.; FRANKS, N.; SNEYD, S.; BONABEAU, E.; THERAULAZ, G. *Self-organisation in biological systems*. Princeton University Press, 2001.
- CASTRO, H. F. *Otimização da Confiabilidade e Disponibilidade em Sistemas Redundantes*. Campinas, 2003. (Mestrado - Faculdade de Engenharia Mecânica, Unicamp).
- CASTRO, L. N.; HRUSCHKA, E. R.; ROSATELLI, M. C. ; CAMPELLO, R. J. G. B. . Computação Natural: Uma Breve Visão Geral. In: *WORKSHOP EM NANOTECNOLOGIA E COMPUTAÇÃO INSPIRADA NA BIOLOGIA*. Rio de Janeiro, 2004.
- CASTRO, L. N.; ZUBEN, F. J. *From biologically inspired computing to natural computing. Recent Developments in Biologically Inspired Computing*. Chapter I, Idea Group Incorporation, pp. 1-8, 2004.
- CHUKOVA, S.; ARNOLD, R.; WANG, D. Q. Warranty analysis: An approach to modeling imperfect repairs. In: *International Journal of Production Economics*, v.89, p.57-68. 2004.
- COELLO, C. A. C.; VELDHUIZEN, D. A. V.; LAMONT, G. B. *Evolutionary algorithms for solving multiobjective problems*. New York: Kluwer Academic, 2002.
- COHEN, P. R. *Empirical methods for artificial intelligence*. MIT Press, Cambridge, Massachusetts, 1995.
- COLORNI A., DORIGO M., MANIEZZO V. Distributed optimization by ant colonies. In: *PROCEEDINGS OF ECAL'91, EUROPEAN CONFERENCE ON ARTIFICIAL LIFE*. Elsevier Publishing, Amsterdam, 1991.
- COOK, W.; LOVÁSZ, L.; SEYMOUR, P. *Combinatorial optimization*. American mathematical society. New Jersey, 1993.
- DEITEL, H. M.; -, P. J. *C++ como programar*.(3ed) Bookman. Porto Alegre, 2005.
- DORIGO, M.; GAMBARELLA, L. M. Ant Colony for the travelling salesman problem. *Biosystems* V. 43, Issue 2, p.73-81, 1997.
- DORIGO, M.; STÜTZLE, T. *Ant Colony Optimization*. Cambridge, Massachusetts: “A Bradford book”, The MIT Press, 2004. 305 p.
- DORIGO, M.; BLUM, C. *Ant colony optimization theory: a survey*. Theoretical Computer Science, 2005.
- DUFFEY, RB. Nuclear power in the 21st century: competitive and environmental imperatives. *Palestra proferida no ANS Latin America section*, Rio de Janeiro, Brasil, 2000.

- EHRGOTT M.; GANDIBLEUX X. A survey and annotated bibliography of multicriteria combinatorial optimization. *OR Spektrum*, forthcoming, 2000.
- ERIKSSON, H. E.; PENKER, M.; LYONS, B.; FADO, D. *UML™ 2 Toolkit*. Wiley Publishing, 2004.
- ERTAS A. *The Engineering Design Process*. John Wiley & Sons Inc, 1993.
- FEBER, J.; GRASSER, L. Intelligence artificielle distribuée. In: *XI International workshop on expert systems & their applications*, Avignon, France, 1991.
- FEOFILOFF, P.; KOHAYAKAWA, Y.; WAKABAYASHI, Y. *Uma introdução sucinta à teoria dos grafos*. 2005. Disponível em < <http://www.ime.usp.br/~pf/teoriadosgrafos/>> acessado em 18 de junho de 2007.
- FOGEL, L. J., OWENS, A. J.; WALSH, M. J. *Artificial intelligence through simulated evolution*. Wiley, New York, 1966.
- FOGEL, D. B. What evolutionary computation? In: *Spectrum*. IEEE, 2000. v. 37, p.26, 28-32
- FONSECA, C. M.; FLEMING, P. J. Genetic algorithms for multi-objective optimization: formulation, discussion and generalization. In: *Proceedings of the Fifth International Conference on Genetic Algorithms*. 1993.
- FRANCATO, A. L.; BARBOSA, P.; Soluções de compromisso na tomada de decisão sobre a operação diária de sistemas urbanos de abastecimento de água. *Revista Brasileira de Recursos Hídricos*, Vol. 9, pp.39-50, Porto Alegre, RS, Brasil, 2004.
- GARCIA, P. A. *Uma abordagem fuzzy com envelopamento dos dados da análise dos modos e efeitos de falha*. Rio de Janeiro, 2006. (Doutorado – Universidade Federal do Rio de Janeiro).
- GRASSÉ, P. La reconstruction du nid et les coordinations inter-individuelles chez *Bellicositermes natalensis* et *Cubitermes* sp. La théorie de la stigmergie: Essai d'interprétation du comportement des termites constructeurs. *Insectes Sociaux*, v.6, p.41-81, 1959.
- GUEDES, G. T. A. *UML – Uma abordagem prática*. (2ª edição). Novatec, 2006.
- HAKEN, H. *Synergetics*. Springer-Verlag, 1977.
- HAKIMI, S.L. Steiner's problem in graphs and its applications. *Networks* vol.1, p. 113-133, 1971.
- HARUNUZZAMAN, M.; ALDEMIR, T. Optimization of standby safety system maintenance scheduling in nuclear power plants. *Nuclear Technology* vol.113, p. 354-367, 1996.

- HOLLAND, J. H. *Adaptation in natural and artificial systems*. MIT Press, 1975.
- HÖLLDOBLER, B.; WILSON, E. *The Ants*. Cambridge, MA: Belknap Press of Harvard University Press, 1990.
- HÜBNER, J. F. *Um modelo de reorganização de sistemas multiagentes*. São Paulo, 2003. 224p. (Doutorado – Engenharia Elétrica, Escola Politécnica da Universidade de São Paulo).
- HÜBNER, J. F.; SICHMAN, J. S. Organização de Sistemas Multiagentes. In: *Congresso da Sociedade Brasileira de Computação, 23.*, 2003. Campinas. Anais. v.8. Campinas: SBC, 2003. p. 247-296.
- KENNEDY, J.; EBERHART, R.; SHI., Y. *Swarm intelligence*. Morgan Kaufmann Publishers, 2001.
- KIJIMA, M.; SUMITA, N. A useful generalization of renewal theory: counting process governed by nonnegative Markovian increments. *Journal of Applied Probability*, v.23, p.71-88. 1986.
- KIRKPATRICK, S.; JR., C.D.G.; VECCHI, M. P. Optimization by simulated annealing. In: *Science*, vol. 220, p. 671-680, 1983.
- KLÜGL, F. *Simulated ant colonies as a framework for evolutionary models*. Universität Würzburg: Dept. of Artificial Intelligence and Applied Computer Science Julius-Maximilians, 2002. 12 p.
- KORTE, B.; VYGEN, J. *Combinatorial optimization: theory and algorithms*. 3º edição. Springer. Berlin, 2002.
- KUO, W.; PRASAD, V. R.; TILLMAN, F. A.; HWANG, C.-L. *Optimal reliability design: fundamentals and applications*. United Kingdom: Cambridge University Press, 2001.
- KURATOWSKI, K. Sur les problème des courbes gauches en topologie. *Fund Math*. vol. 15, p. 271-283, 1930.
- LAFRAIA, J. R. B. *Manual de confiabilidade, manutenibilidade e disponibilidade*. Qualitymark: Petrobras. Rio de Janeiro, 2001.
- LAPA, C. M. F.; PEREIRA, C. M. N. A.; BARROS, M. P. de. A model for preventive maintenance planning by genetic algorithms based in cost and reliability. In: *Reliability Engineering & System Safety*, vol. 91, p. 233–240, 2006.
- LAWLER, E. L.; LENSTRA, J. K.; RINNOOY, A. H. G. *The traveling salesman problem : a guided tour of combinatorial optimization*. Chichester: Wiley, 1985.

- LEGAT, V.; ZALUDORA, A.H.; CERVENKA, V.; JURCA, V. Contribution to optimization of preventive replacement. In: *Reliability Engineering & System Safety*, vol. 51, p.259-266, 1996.
- LEVITIN, G.; LISNIANSKI, A. Optimization of imperfect preventive maintenance for multi-state systems. . In: *Reliability Engineering & System Safety*, vol. 67, p. 193-203, 2000.
- LEVITIN, G. *Computational intelligence in reliability engineering*. Israel: Springer, 2007.
- LEWIS, E.E. *Introduction to reliability engineering*. John Wiley and Sons, Inc., New York, 1996.
- MARSEGUERRA, M.; ZIO, E. Optimizing maintenance and repair policies via a combination of genetic algorithms and Monte Carlo simulation. In: *Reliability Engineering & System Safety*, vol. 68, p. 69–83, 2000.
- MARSEGUERRA, M.; ZIO, E.; BUSACCA, P. Multiobjective optimization by genetic algorithms: application to safety systems. In: *Reliability Engineering & System Safety*, vol. 72, p. 59–74, 2000.
- MARSEGUERRA, M.; ZIO, E.; PODOFILLINI, L. Multiobjective spare part allocation by means of genetic algorithms and monte carlo simulation. In: *Reliability Engineering & System Safety*, vol. 87, p. 325–335, 2005.
- MARSEGUERRA, M.; ZIO, E.; MARTORELL, S. Basics of genetic algorithms optimization for RAMS applications. In: *Reliability Engineering & System Safety*, vol. 91, p. 977–991, 2006.
- MCCULLOCH, W.; PITTS, W. *A logical calculus of ideas immanent in nervous activity*. Bulletin of Mathematical Biophysics, 1943.
- MODARRES, M.; KAMINSKIY, M.; KRIVTSOV, V. *Reliability Engineering and Risk Analysis*. New York, Marcel Dekker, Inc., 1999.
- MODARRES, M. *Risk analysis in engineering: techniques, tools and trends*. Boca Raton: Taylor & Francis, 2006.
- MOREIRA, D. H; SABOIA, P.C. *Sistemas multiagentes baseado em colônia de formigas*. Belém, 2006. (Bacharel – Ciência da computação, UFPA).
- MOURA, M. J.; DROUGUETT, E. L. Determinação do grau de eficácia de equipes de manutenção via Processo de Renovação Generalizado. In: *XXVI ENEGEP*, 2006.
- NARASIMHALU, A. D.; SIVARAMAKRISHNAN, H. A Rapid Algorithm for Reliability Optimization of Parallel Redundant System. In: *IEEE Transactions on Reliability*, v. R-27, n. 4, pp. 261-263, 1978.

- NICOLIS, G.; PRIGOGINE, I. *Self-organization in non-equilibrium systems*. Wiley, 1977.
- PALAZZO, L. A. M. Complexidade, caos e auto-organização. In: *Oficina de Inteligência Artificial*. vol. 3. p. 49-67. Pelotas, 1999.
- PAPADIMITRIOU C.H.; STEIGLITZ K. *Combinatorial Optimization—Algorithms and Complexity*. Dover Publications Inc., New York, 1982
- PELLERINI, P; DORIGO, M. On the invariance of ant colony optimization. In: *IEEE Transactions on evolutionary computation*. v. 11, no.6, p.732-742, 2007.
- PENDER, T. *UML – A Bíblia*. Campus/Elsevier, 2004.
- PEREIRA, C; LAPA, C. Parallel island genetic algorithm applied to a nuclear power plant auxiliary feedwater system surveillance tests policy optimization. In: *Annals of Nuclear Energy*. Vol. 30 p.p. 1665–1675, 2003.
- RAUSAND, M.; HOYLAND, A. *System reliability theory: models and statistical methods*. 2ed. New York: John Wiley & Sons, 2003.
- RECHENBERG, I. *Evolutionsstrategie: optimierung technischer systeme nach prinzipien der biologischen evolution*, Frommann-Holzboog, Stuttgart, 1973.
- RIBEIRO, P. C. *Modelagem e implementação OO de sistemas multi-agentes*. Rio de Janeiro, 2001. 106p. (Mestrado - Pontifícia Universidade Católica do Rio de Janeiro).
- RICARTE, I. L. M. *Programação orientada a objetos com C++*. Unicamp, 2001. Disponível em < http://www.dca.fee.unicamp.br/cursos/POO_CPP/POO_CPP.html > acessado em 01 de julho de 2007
- RIGDON, S. E.; BASU, A. B. *Statistical methods for the reliability of repairable systems*. John Wiley & Sons, 2000.
- ROSS, S. M. *Stochastic process*. New York: John Wiley.1983.
- ROSS, S. M. *Introduction to probability models*. 7ed. San Diego: Academic Press, 2000.
- RUSSEL, S.; NORVIG, P. *Artificial Intelligence: a modern approach*. New Jersey: Prentice-Hall, 1995.
- RUSSEL, S.; NORVIG, P. *Inteligência Artificial*. Editora Campus. Rio de Janeiro, 2004.
- SAMROUT, M; YALAOUI, F; CHÂTELET, E; CHEBBO, N. New methods to minimize the preventive maintenance cost of series-parallel systems using ant colony optimization. In: *Reliability Engineering & System Safety*, vol. 89, p. 346–354, 2005.

- SARAMAGO, S; PRADO, J. Otimização por colônia de partículas. *XXVIII Congresso Nacional de Matemática Aplicada e Computacional*, 2005.
- SASAKI, M.; KABURAKI, S.; YANAGI, S. System availability and optimum spare units. In: *IEEE Transactions on Reliability*, v. R-26, n. 3, pp. 182-188, 1977.
- SCHWEFEL H.P. *Evolution and Optimum Seeking*. John Wiley & Sons Inc, United States of America, pp. 87-88, 1994.
- SEDGEWICK, R. *Algorithms in C*. 3ª edição. Addison-Wesley , 2000.
- SILVA, I. G. L. *Projeto e Implementação de Sistemas Multi-Agentes: O Caso Tropos*. Recife, 2005. p.108 (Mestrado - UFPE).
- SOCHA, K.; SAMPLES, M; MAFRIN, M. Ant algorithms for the university course timetabling problem with regard to the state-of-art. In: *3rd European workshop on evolutionary computation in combinatorial optimization*. v. 2611, p. 334-345, 2003.
- SOCHA, K.; DORIGO, M. Ant colony optimization for mixed-variable optimization problems. In: *IRIDIA. No. TR/IRIDIA/2007-019*, 2007.
- SOCHA, K.; DORIGO, M. Ant colony optimization for continuous domains. *European Journal of Operational Research*. v. 185, p. 1155-1173, 2008.
- SOUZA, M. J. F. *Modelagens exata e heurística para resolução do problema do caixeiro viajante com coleta de prêmios*. Ouro Preto, 2003.(Monografia – Universidade Federal de Ouro Preto).
- TABOADA, H. A.; COIT, D. W. Multiple objective design allocation problems: development of new evolutionary algorithms. In: *Proceedings of the European Safety & Reliability Conference (ESREL)*. Stavanger, Norway: 2007.
- TILLMAN, F. A.; HWANG C. L.; KUO, W. Optimization techniques for system reliability with redundancy – a review. In: *IEEE Transactions on Reliability*, v. R-26, n. 3, pp. 148-155, 1977.
- TRIVEDI, K. S. *Probability and statistics with reliability, queuing, and computer science applications*. John Wiley and Sons, Inc., New York, 2002.
- TSAI, Y. T.; WANG, K. S.; TENG, H. G. Optimizing preventive maintenance for mechanical components using genetic algorithms. In: *Reliability Engineering & System Safety*, vol. 74, p. 89–97, 2001.
- TURING, A. M. *Computing Machinery Intelligence*. Oxford university press, v. LIX, n.236, p.433-60, 1950.

- VITTORI, K. *Estudo experimental, modelagem e implementação do comportamento de colônias de formigas em um ambiente dinâmico*. São Carlos, 2005. (Doutorado – Escola de Engenharia de São Carlos, Universidade de São Paulo).
- WACKERLY, D. D.; MENDENHALL, W.; SCHEAFFER, R. L. *Mathematical Statistics with Applications*. Duxbury Press, 1996.
- WEST, D. B. *Introduction to graph theory*. 2ª edição. Prentice Hall, 2001.
- WILSON, E. O. *The insect societies*. Cambridge, MA: Belknap Press of Harvard University Press, 1971.
- WOHL, J. G. System operational readiness and equipment dependability. In: *IEEE Transactions on Reliability*, v. R-15, n. 1, pp. 1-6, 1966.
- YAÑES, M.; JOGLAR, F.; MODARRES, M. Generalized renewal process for analysis of repairable systems with limited failure experience. In: *Reliability Engineering & System Safety*, vol. 77, p. 167–180, 2002.
- YOON, K. P.; HWANG, C.-L. *Multiple attribute decision making: an introduction*. Thousand Oaks: Sage, 1995.
- ZITZLER, E. *Evolutionary algorithms for multiobjective optimization: methods and applications*. Zurich, 1999 (Doutorado —Swiss Federal Institute of Technology Zurich).
- ZHAO, J.; LIU, Z.; DAO, M. Reliability optimization using multiobjective ant colony system approaches. In: *Reliability Engineering and System Safety*, vol. 92. p. 109–120, 2007.
- ZOMAYA, A.Y. Natural and simulated annealing. In: *Computing in Science & Engineering*, vol.3, p.97-99, 2001.

O48e

Oliveira, Rosana Cavalcante de.

Otimização multiobjetivo da confiabilidade via sistemas multiagentes baseado em colônia de Formigas / Rosana Cavalcante de Oliveira. - Recife: O Autor, 2008.

xi, 71 folhas, il : figs., tabs., gráfs.

Dissertação (Mestrado) – Universidade Federal de Pernambuco. CTG. Programa de Pós-Graduação em Engenharia de Produção, 2008.

Inclui Bibliografia.

1. Engenharia de Produção. 2. Engenharia de Confiabilidade. 3. Sistemas Multiagentes. I. Título.

UFPE

658.5

CDD (22. ed.)

BCTG/2008-089

Livros Grátis

(<http://www.livrosgratis.com.br>)

Milhares de Livros para Download:

[Baixar livros de Administração](#)

[Baixar livros de Agronomia](#)

[Baixar livros de Arquitetura](#)

[Baixar livros de Artes](#)

[Baixar livros de Astronomia](#)

[Baixar livros de Biologia Geral](#)

[Baixar livros de Ciência da Computação](#)

[Baixar livros de Ciência da Informação](#)

[Baixar livros de Ciência Política](#)

[Baixar livros de Ciências da Saúde](#)

[Baixar livros de Comunicação](#)

[Baixar livros do Conselho Nacional de Educação - CNE](#)

[Baixar livros de Defesa civil](#)

[Baixar livros de Direito](#)

[Baixar livros de Direitos humanos](#)

[Baixar livros de Economia](#)

[Baixar livros de Economia Doméstica](#)

[Baixar livros de Educação](#)

[Baixar livros de Educação - Trânsito](#)

[Baixar livros de Educação Física](#)

[Baixar livros de Engenharia Aeroespacial](#)

[Baixar livros de Farmácia](#)

[Baixar livros de Filosofia](#)

[Baixar livros de Física](#)

[Baixar livros de Geociências](#)

[Baixar livros de Geografia](#)

[Baixar livros de História](#)

[Baixar livros de Línguas](#)

[Baixar livros de Literatura](#)
[Baixar livros de Literatura de Cordel](#)
[Baixar livros de Literatura Infantil](#)
[Baixar livros de Matemática](#)
[Baixar livros de Medicina](#)
[Baixar livros de Medicina Veterinária](#)
[Baixar livros de Meio Ambiente](#)
[Baixar livros de Meteorologia](#)
[Baixar Monografias e TCC](#)
[Baixar livros Multidisciplinar](#)
[Baixar livros de Música](#)
[Baixar livros de Psicologia](#)
[Baixar livros de Química](#)
[Baixar livros de Saúde Coletiva](#)
[Baixar livros de Serviço Social](#)
[Baixar livros de Sociologia](#)
[Baixar livros de Teologia](#)
[Baixar livros de Trabalho](#)
[Baixar livros de Turismo](#)