ROBERTO FERNANDES TAVARES NETO

PLANEJAMENTO DE VISTORIAS USANDO ROBÔS MÓVEIS AUTÔNOMOS E OTIMIZAÇÃO PELO ALGORITMO DE COLÔNIA DE FORMIGAS

Dissertação apresentada ao Programa de Pós-Graduação em Engenharia de Produção e Sistemas da Pontifícia Universidade Católica do Paraná como requisito parcial para obtenção do título de Mestre em Engenharia de Produção e Sistemas.

CURITIBA

2005

ROBERTO FERNANDES TAVARES NETO

PLANEJAMENTO DE VISTORIAS USANDO ROBÔS MÓVEIS AUTÔNOMOS E OTIMIZAÇÃO PELO ALGORITMO DE COLÔNIA DE FORMIGAS

Dissertação apresentada ao Programa de Pós-Graduação em Engenharia de Produção e Sistemas da Pontifícia Universidade Católica do Paraná como requisito parcial para obtenção do título de Mestre em Engenharia de Produção e Sistemas.

Área de Concentração: Automação e Controle

Orientador: Prof. Dr. Leandro dos Santos Coelho

CURITIBA

2005

Tavares Neto, Roberto Fernandes
Planejamento de Vistorias Usando Robôs Móveis Autônomos e Otimização pelo Algoritmo de Colônia de Formigas. Curitiba, 2005. 99p

Dissertação – Pontifícia Universidade Católica do Paraná. Programa de Pós-Graduação em Engenharia de Produção e Sistemas.

1. Inspeção 2. Manutenção Preventiva 3. Colônia de Formigas 4. Otimização. I.Pontifícia Universidade católica do Paraná. Centro de Ciências Exatas e de Tecnologia. Programa de Pós-Graduação em Engenharia de Produção e Sistemas.

Agradecimentos

Ao professor Leandro dos Santos Coelho pela inspiração, orientação, competência e paciência durante todo o desenvolvimento desta dissertação.

À PUCPR pelo incentivo à pesquisa no período em que fui bolsista de mestrado, e a todos os envolvidos nos projetos em que tive o prazer de trabalhar, pela fé em minha capacidade.

À minha irmã Ana Flávia, pelo carinho e apoio nas horas ruins e pelos momentos compartilhados nas horas boas.

Ao meu irmão Tiago, pela ajuda no decorrer da dissertação.

E, em especial, à minha mãe, que me surpreende a cada dia com seu complexo conhecimento da humanidade, ao meu pai, por ter me mostrado as potencialidades da engenharia, e a ambos, por compartilhar uma compreensão do mundo que me permite hoje fazer o bem.

A todos que, direta ou indiretamente, colaboraram para a realização deste trabalho.

Sumário

Lista de figuras

Lista de Tabelas

Lista de siglas

ACS – *Ant Colony System* – Sistema de Colônia de Formigas
AGVs – *Automated Guied Vehicles* – Veículos Guiados Automaticamente
ROV – *Remoted Operated Vehicle* – Veículo Operado Remotamente
FIRA - *Federation of International Robot-soccer Association* – Associação Internacional de Futebol de Robôs
GA – *Genetic Algorithm* – Algorítimo Genético
AG – Algoritmo Genético
SA – *Simulated Annealing*
PE – Programação Evolutiva
UAV - *Unmanned Aerial Vehicles* – Veículos aéreos não tripulados
BEES – *Bioinspired Engineering of Exploration Systems* – Sistemas de exploração inspirados biologicamente
TSP – *Travelling Salesman Problem* – Problema do caixeiro viajante
DA – *Deterministic Annealing*
RNA – Redes neurais artificiais
QAP – *Quadratic Assighment Problem* – Problema da designação quadrática
JSP – *Job-shop Scheduling Problem* – Problema da escalonamento de turnos
FILO – *First in, Last Out* – Primeiro a entrar, último a sair.

Resumo

A inspeção robotizada em locais de difícil acesso é um problema cada dia mais presente no cotidiano. Na busca de soluções a este desafio, observa-se o surgimento de iniciativas, seja de forma orientada como nas competições robóticas a exemplo do *Robocup Rescue*, seja em aplicações práticas em que companhias oferecem veículos de operação remota para inspeção de tubulações, galerias e demais estruturas subterrâneas. A seqüência natural da evolução destes serviços, conforme já notado por diversos pesquisadores, é a automação destas tarefas. Com esta automação, os robôs podem realizar, automaticamente, vistorias em locais de difícil acesso ou que oferecem riscos à integridade humana, de forma contínua, sem exigir a presença constante do ser humano no controle e avaliação da situação. Este trabalho apresenta um conjunto de heurísticas de um sistema multi-agente baseadas em reforços positivos gerados através de comunicação sinergética. Os negativos são baseados na memória dos agentes e modificações no espaço de busca através de algoritmos culturais para a automação do planejamento de inspeções. Através de modificações realizadas no algoritmo de inteligência coletiva, baseado no comportamento de colônia de formigas, mostra-se a viabilidade dos serviços a serem executados de forma autônoma, mesmo em situações difíceis, tais como aquelas em que os robôs necessitam de cordões umbilicais para envio de dados e alimentação dos motores elétricos. Buscando o aperfeiçoamento das heurísticas apresentadas, detectadas através do aumento da rapidez da convergência do algoritmo, esta dissertação se utiliza do paradigma denominando Algoritmo Cultural. Através da adaptação de heurísticas com esta inspiração, é possível à colônia a redução do espaço de busca, permitindo que o esforço de otimização seja convergido para soluções de um mínimo local do problema, convergência não encontrada no algoritmo original.

Palavras-chave: Inspeção, Manutenção Preventiva, Colônia de Formigas, Otimização.

Abstract

Robot-based inspection on sites with difficult access is a problem that becomes more common nowadays. Searching for solutions to this challenge, it's observed new initiatives, either in orientated form in robot competitions like Robocup Rescue, or practical applications in companies that offer remoted operated vehicles to inspect pipes, galleries and further underground facilities. The natural evolution sequence for those services, as mentioned by several researches, is the automation of those tasks. With this automation, robots can accomplish, in an independent form, inspections of hard-access or dangerous places continually, without requiring the constant operation by a human being. This dissertation presents a set of heuristics of a multi-agent system based on positive reinforcements generated by a synergetic communication and negative reinforcements based on individual agent memory and modifications at the searching area using cultural algorithms to the automation of inspections. Beyond modifications on the collective intelligence, based on ant colonies behavior, it's presented in this document the viability of the automation of this services, even on hard situations such as robots that require umbilical cables for data communication and power supply. Looking for improvement to the presented heuristics, detected by the of the algorithm convergence, this dissertation uses the Cultural Algorithms paradigm. Beyond adaptations of heuristics with this inspiration, the colony could reduce the searching space, allowing the optimization effort to be converted to the solution of a reduced part of the problem, this convergence could not be observed on the original algorithm.

Keywords: Inspection, Preventive Maintanence, Ant Colony, Optimization.

Capítulo 1. Introdução

Vários setores de prestação de serviços na área de construção, como aqueles relacionados com a manutenção preventiva e corretiva, já definiram como necessária a inspeção de locais de difícil acesso pelo ser humano. Lockheed (1997) afirma que "existe uma necessidade no Departamento de Energia [dos Estados Unidos] em inspecionar as condições internas de tubulações e outros ambientes inacessíveis, perigosos ou restritos durante a descontaminação e decomissionamento (desativação) de instalações sem uso". Em seu relato técnico de abril de 2001, a *New York Gas Group* afirma que "durante a década passada, a inspeção robótica tem encontrado um aumento de uso em instalações de gás" (New York Gas Group, 2001).

A Paul Hayward Associates (2003) afirma que "grande parte do sistema de esgoto do Reino Unido e de outros países industrializados se encontram em precário estado de conservação." como pode ser visto na Fig. 1.1.

Figura 1.1 Exemplo de tubulação em uso, em pobre estado de conservação (fonte: Paul Hayward Associates).

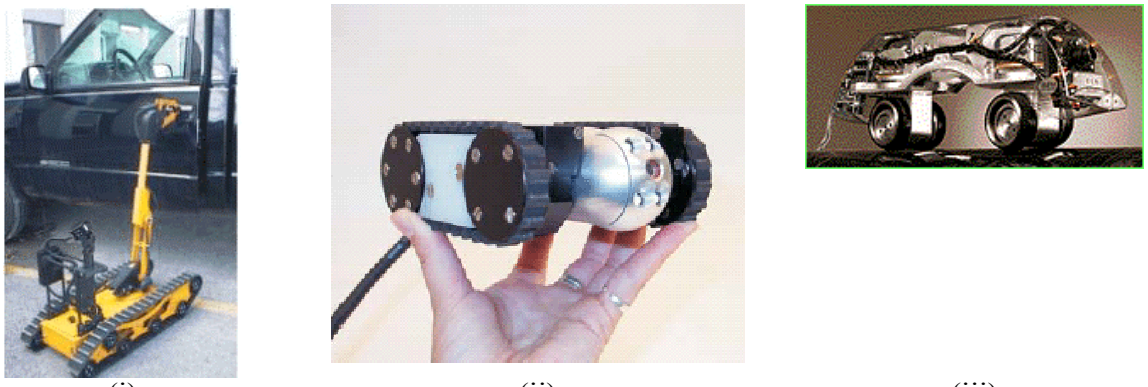
Com o objetivo de evitar os prejuízos devido ao efeito do envelhecimento de encanamentos, organizações como o Departamento de Energia Americano têm criado programas de pesquisa visando garantir a segurança e a eficiência operacional de seus serviços (Panetta *et al*., 2001).

Venugopal (2001) enumera uma gama de testes não destrutíveis empregados pela indústria e tendências atuais de pesquisa na área de inspeção de tubulações. Estas são:

- detecção de falhas: as causas das falhas de um sistema de tubulação são várias, tais como a corrosão e dano por ações de equipamentos de obras;
- testes não destrutivos baseados no vazamento de fluxo magnético;
- testes não destrutivos baseados no método ultrasônico.

De acordo com Panetta *et al.* (2001), os testes mais usados para a detecção de falhas mecânicas são os que se utilizam de ferramentas de inspeção *in-line*. Como inspeção *in-line*, entende-se sistemas de instrumentação que se movem de um ponto a outro no interior de uma tubulação.

A inspeção através de veículos controlados remotamente (*remote controlled vehicles* – ROVs) já é uma realidade. Várias empresas, como a Roboprobe e a SRI International, oferecem ao mercado diversos sistemas teleoperados para inspeção. Alguns exemplos são mostrados na Figura 1.2.



(i) (ii) (iii)

Figura 1.2 Exemplo de ROVs. (i) um ROV para desarmamento de bombas; (ii) e (iii), ROVs para inspeção de tubulações (fontes: RoboProbe Technologies Inc., 2003; SRI International, 2003).

Tais tarefas de inspeção requerem, atualmente, um controlador de superfície para monitorar e operar o ROV. Tais tarefas costumam ser repetitivas, pois a topologia da construção a ser monitorada não tende a mudar constantemente.

O estudo da automatização destes sistemas de forma que a ação humana seja mínima é uma realidade (Byrne *et al*., 2002; Jianxu *et al.*, 2000). Whittaker *et al.* (2000) mostra resultados na automação de montagens, inspeções e manutenção de estruturas em aplicações espaciais. Estudos neste contexto, que permitem aos robôs realizarem tarefas em ambientes complexos e dinâmicos, são desafios importantes no estudo da robótica (Kita *et al.*, 1999).

Supondo que esta tecnologia – a inspeção robotizada – seguirá o rumo de várias outras, é possível supor que logo esta automação estará presente no dia-a-dia, como mais uma forma de prestação de serviço.

Como toda prestação de serviço de manutenção, existem variáveis cuja alteração afetam o custo do serviço:

- o tempo de duração da inspeção, que gera custos tanto relacionados à alocação da equipe que acompanha a inspeção quanto de possíveis paradas do sistema a ser verificado (no caso, por exemplo, de paradas em usinas hidroelétricas);
- o desgaste de equipamento durante o uso, presente em qualquer sistema mecânico.

Para a minimização de ambas as variáveis, um dos fatores a ser perseguido é a otimização do caminho a ser percorrido pela inspeção.

Este problema também estará presente em versões autônomas destes sistemas de inspeção, que quando automatizados, ganham maior importância. Citando Nehmzow (2001): "aplicações industriais de robôs móveis tem ganho importância continuamente, em particular sob considerações de confiabilidade (operação ininterrupta de tarefas monótonas como vigilância), acessibilidade (inspeção de locais que são inacessíveis por humanos, tais como espaços estreitos, ambientes hostis ou remotos) e custo (sistemas autônomos de transporte podem ser mais baratos que formas tradicionais de transporte)".

Kita *et al.* (1999) mostra a aplicação de um sistema composto de dois robôs autônomos atuando em conjunto para a inspeção em usinas nucleares.

Nehmzow (2001) indica a importância de um controle adaptativo em um robô móvel, evitando o que o autor chamou de "*hard-wired*" (inflexível), sendo este indicado como sendo a forma preferida de implementação para ambientes dinâmicos. Com o controle adaptativo, ou seja, com a habilidade do robô móvel de aprender, mudanças na sua percepção de mundo podem ser toleradas, levando o sistema de controle a se adequar à mudança do mundo, da tarefa, ou de sua própria morfologia.

Alami *et al.* (2002) notam também a necessidade de uma autonomia avançada para sistemas embarcados em tempo real, como robôs, satélites e aeronaves não tripuladas (UAV – *Unmanned Aerial Vehicles*).

Koening (1999) mostra alguns algoritmos de planejamento através de sistemas centrados em agentes simples ou sistemas multi-agentes.

Thakoor *et al.* (2002) descrevem algumas saídas encontradas por insetos para executar tarefas que, embora simples para humanos, ainda têm sido alvo de pesquisas no desenvolvimento de vários robôs. Tendo como foco o vôo, o reconhecimento de obstáculos ou o pouso de insetos como abelhas e libélulas, os autores mostram como sistemas de

exploração inspirados na biologia (BEES – *Bioinspired Engineering of Exploration Systems*) podem ser utilizados na automação de robôs da NASA.

A idéia de planejamento de tarefas através de algoritmos inspirados em BEES, também é explorada por Bonabeau e Theuralaz (1998). Neste artigo, os sistemas inspirados em comportamentos de enxames de insetos são utilizados para se obter soluções de problemas de caixeiro viajante, roteamento de veículos, transporte cooperativo e agendamento de tarefas.

Uma característica presente em grande parte destes algoritmos é a possibilidade de ser executado utilizando-se de processamento paralelo, normalmente de complexidade e custo computacional menor que o algoritmo como um todo. Esta caracteristica foi utilizada por Schoonderwoerd *et al*. (1996), que apresenta um modelo de rede populado por formigas artificiais para o roteamento de chamadas de uma rede de comunicação heterogênia e dinâmica. Neste caso, as formigas depositam "feromônios" artificiais (usados como um reforço positivo de parâmetros de um algoritmo pseudo-aleatório) de acordo com o congestionamento do nó e o comprimento do caminho da formiga. Sim e Sun (2003) analisam o uso de algoritmos de otimização por colônia de formigas (ACS – *Ant Colony System*) e comparam seu uso com métodos tradicionais de roteamento de chamadas. São destacados três aspectos:

- Para algoritmos tradicionais, é necessário o conhecimento de toda a rede para a construção da tabela de roteamento (com informações sobre topologia da rede e respectivo tráfego nos roteadores). No caso do ACS, cada "formiga" apenas se utiliza da informação dos nós adjacentes. Com isso, cada transmissão de pacotes de um elemento roteador da rede para outro é uma instância independente, sendo suas propriedades (quantidade de feromônio) atualizadas em função da chegada ou não de formigas no roteador. Ao contrário da abordagem tradicional, um roteador de uma rede que se utilize de um algoritmo como o ACS apenas necessita conhecer as relações com os roteadores adjacentes, e sua operação consiste apenas em atualizar feromônios e redirecionar pacotes segundo uma função pseudo-aleatória simples.
- Ainda em algoritmos tradicionais, o roteamento de chamadas requer a transmissão de tabelas de roteamento para todos os nós da rede. O tamanho desta tabela é proporcional ao tamanho da rede. O uso de algoritmos baseados

em ACS requerem apenas que um pequeno código (a "formiga") seja enviada pela rede. De acordo com Sim e Sun (2003), este pedaço de código tem um tamanho na ordem de 6 bytes. O tráfego resultante deste pequeno código é em muito superado quando confronta-se o tamanho de transmissões de dados realizados diariamente, como acesso à sites WEB populares como o Google (11.776 bytes) (Google, 2004), e portais que possuem grande volume de acessos diários, como o Universo Online (114.176 bytes) (UOL, 2004) e o Terra (70.144 bytes) (Terra, 2004).

- Para evitar a sobrecarga da rede ao retransmitir de forma excessiva a tabela de roteamento em algoritmos tradicionais, pode-se escolher aumentar o intervalo de transmissão da mesma. Porém, esta estratégia falha ao aumentar o tempo pelo qual os padrões de roteamento se manterão, aumentando o tempo de resposta à mudanças da topologia da rede. Com o uso de ACS, o pequeno trecho referente à formiga não causa transtornos significativos à rede, sendo que seu intervalo de transmissão pode ser reduzido, tornando este algoritmo sensível à mudanças na rede, sejam elas de topologia (roteador desligado) ou de tráfego (canais de transmissão sobrecarregados).

Existem algumas características interessantes na aplicação de sistemas inspirados em formigas para o roteamento de redes. Inicialmente, pode-se caracterizar esta aplicação como um sistema multi-agente (Franklin e Graesser, 1996), aonde não há interação direta entre os integrantes do sistema: em vez disso, a comunicação se dá através do meio, pelo depósito e percepção de feromônios (comunicação indireta). A complexidade deste sistema, se dá não pela ação de um único elemento, mas sim desta interação, que proporciona ao conjunto de agentes resolver problemas cuja solução não seria alcançável por um único elemento (Zhong e Evans, 2002).

Este conecito é utilizado na robótica por várias pesquisas. Dorigo et al. (2004), mostra um estudo de caso na formação de comportamentos complexos como a movimentação coordenada e agregação de agentes. Através do conceito de enxames de robôs (*swarm robots*) introduzido pelos pesquisadores, conseguiu-se alguns resultados promissores. Nos experimentos apresentados por Dorigo *et al.* (2004), um conjunto de *s-bots* – robôs autônomos relativamente simples de reduzido poder computacional, de ação e

sensoriamento limitado – foram capazes de formar um enxame de robôs (*swarm-bot*). Estes experimentos demonstraram que a evolução do *swarm-bot* foi capaz de produzir um sistema auto-organizável baseado em regras gerais e simples, resultando em um sistema robusto e escalável (Dorigo *et al.*, 2004).

Percebendo o potencial de heurísticas inspiradas em enxames e colônias, esta dissertação visa o estudo, a caracterização e a implementação de um sistema bio-inspirado, adaptando heurísticas já estabelecidas na literatura ao problema de planejamento de rotas de ROVs, e aumentando seu desempenho através da combinação de algoritmos culturais (Reynolds e Chung, 1997; Iacoban *et al*., 2003; Coello e Becerra, 2003) com ACS.

A dissertação está organizada da seguinte forma. Após a introdução, no capítulo 2, o problema é descrito com detalhes e são apontadas as limitações físicas previstas. No capítulo 3 é apresentada uma introdução à robótica móvel. No capítulo 4 são apresentados métodos de resolução do problema, enfatizando o método de ACS. No capítulo seguinte, são apresentados e discutidos os estudos de casos realizados. Por fim, no capítulo 6 são mencionadas as conclusões e perspectivas de futura pesquisa.

Capítulo 2. Descrição do problema

Conforme mencionado no capítulo anterior, o foco desta dissertação é o planejamento de uma rota de inspeção para um robô disponível no mercado para trabalhar em dutos de médio calibre (maiores de 150 mm). Para isso, inicia-se definindo o problema de planejamento de caminhos para a robótica autônoma.

Thomaz *et al*. (1999) formularam o problema de planejamento de caminho de um robô autônomo como sendo "dado um robô e o ambiente no qual ele está inserido, planejar um caminho entre dois locais específicos que seja livre de colisões e satisfaça alguns critérios de otimização especificados".

Para abordar este problema, é utilizado o conceito de sistema auto-organizável definido por Ünsal (1993). De acordo com Ünsal (1993), um sistema auto-organizável é um sistema com três características principais:

- "affect", ou seja, consegue analisar o meio no qual o robô está inserido. Esta característica trata das limitações que o meio impõe ao robô móvel, assim como as conseqüências que tais limitações trazem no projeto do mesmo;
- "telos" (do grego, aqui entendido como sendo "avaliação") que representa a capacidade de planejamento de uma ação;
- "efeito" que simboliza as ações tomadas pelo sistema, entre as quais acionamento de motores, ajuste de foco de câmeras, controle de luminosidade de lâmpadas e outros atuadores do robô móvel.

2.1. Limitações físicas do meio

A princípio, supõe-se que existe o conhecimento da planta dos caminhos a serem inspecionados. O tempo esperado para a inspeção de cada trecho deve ser um parâmetro conhecido *a priori*. No entanto, é razoável supor que tais variáveis mudem ao longo da inspeção. São esperados, em quaisquer inspeções deste tipo, que se encontrem quebras na tubulação (Figura 2.1(a)), infiltrações (Figura 2.1(b)) e até mesmo corpos estranhos, como as raízes de plantas (Figura 2.1(c)).

(i) (ii) (iii)

Figura 2.1 Problemas encontrados durante uma inspeção de instalações. Da esquerda para a direita: (i) quebra nas tubulações; (ii) infiltrações; (iii) raízes dentro da tubulação (fonte: Metropolitan Sewer District, 2003).

No contexto desta pesquisa, supõe-se que os efeitos do meio possíveis são apenas no que tangem ao impedimento do movimento do ROV. Outros estudos como efeito de fluxo de água no ROV, e outras possíveis interferências do meio sobre sua operação, como por exemplo, a retenção da estrutura mecânica, são tópicos de estudos posteriores.

2.2. Limitações físicas do ROV

Grande parte dos ROVs disponíveis comercialmente pesquisados (Universidade de Oldenburg (2003), RoboProbe (2003), SRI International (2003)) possuem uma boa manobrabilidade. Normalmente, robôs de tubulações são compostos por duas esteiras que permitem a realização de movimentos em curvas.

Existem outras morfologias possíveis de robôs para esta finalidade, como robôs peristálticos (Jianxu *et al.*, 2000). Tais robôs, ainda em fase de pesquisa e portanto não estão disponíveis comercialmente, não são alvos desta pesquisa. Porém, não foram encontrados indícios que impeçam a extensão dos resultados aqui obtidos para novas morfologias com suas devidas condições de contorno impostas por diferentes configurações mecânicas.

Sendo assim, a mobilidade do robô, ou mais especificamente a impossibilidade de que este realize curvas ou seja impedido de se movimentar não são fatores considerados obstáculos à resolução do problema apresentado. Porém, também lhes é compartilhado uma limitação: o cabo umbilical, mostrado na Figura 2.2 (Watanasin *et al.*, 2001). Este cabo,

necessário para a transmissão de sinais de controle e alimentação, assim como para a recepção dos dados dos sensores do ROV, possui um comprimento finito que limita o planejamento das ações do mesmo. Assim, o problema de planejar a inspeção toma um novo contorno, ao ser verificado que este cabo se move apenas com o auxílio do ROV, sendo para o ROV um limitante da distância percorrida.

O cabo umbilical dificilmente será removido dos ROVs nos próximos anos. Existem vários motivos para isso: primeiro, os ROVs normalmente se utilizam de motores elétricos, que requerem uma quantidade considerável de armazenadores elétricos (baterias), com consequente aumento do peso do ROV (Watanasin *et al.*, 2001). Segundo, o armazenamento de imagens das inspeções não é realizado no ROV por questões de espaço físico, peso e segurança dos dados. Este envio de informações deve ser feito através de cabos pois, muitas vezes, o meio interfere nas comunicações de rádio. Por fim, é interessante a disponibilidade de alguma forma de recuperar o ROV no caso do sistema ficar sem reação em algum ponto (imobilidade devido a obstáculos no ambiente, problemas mecânicos etc).

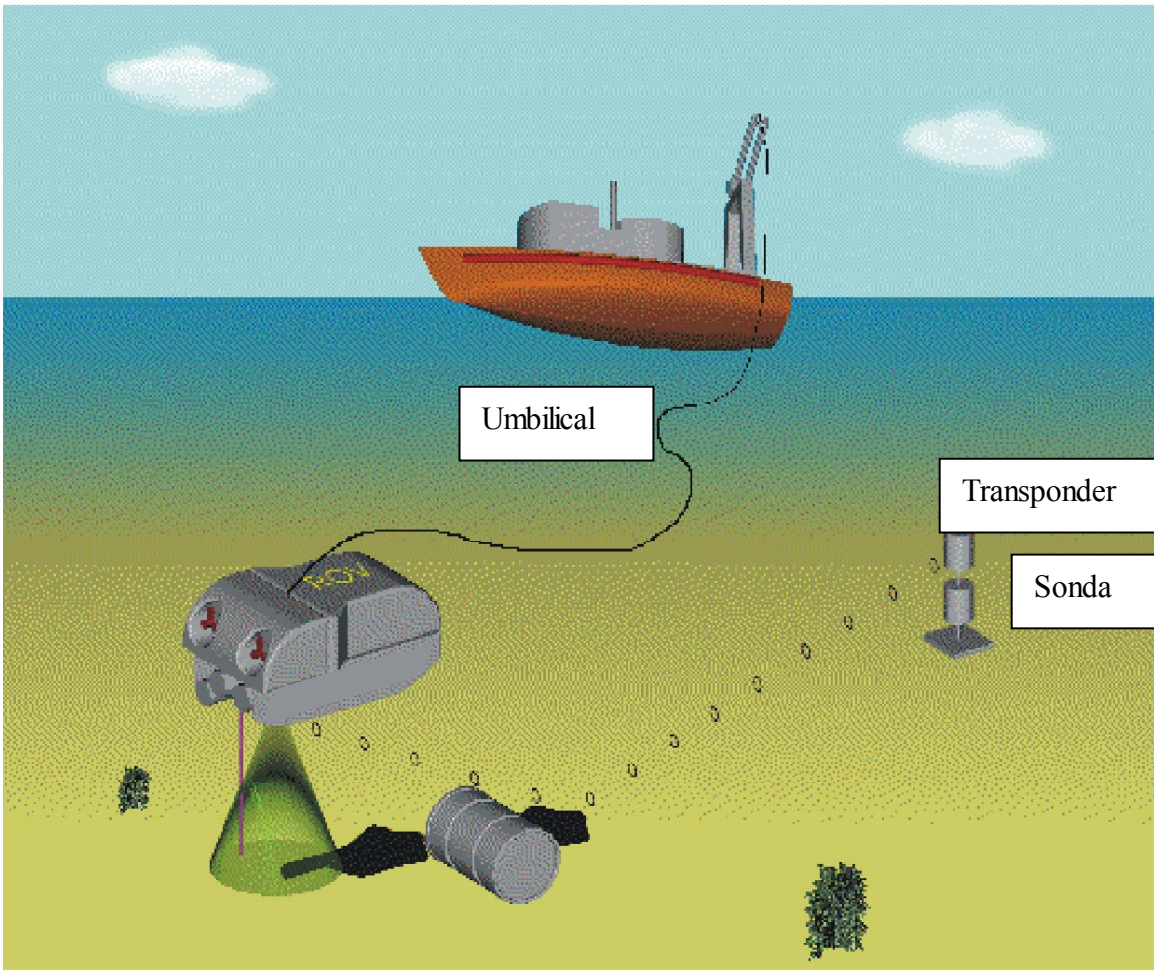


Figura 2.2 Exemplo de ROV ligado à base de operações através de um cabo umbilical (Fonte: Universidade de Oldenburg, 2003).

2.3. Análise de arquiteturas para autonomia de robôs

Alami *et al.* (1998) enumeram seis propriedades requeridas por uma estrutura de controle de robôs:

1. Programabilidade – um robô não deve ser concebido apenas para uma tarefa ou ambiente descrito em detalhes, mas suas funções devem ser facilmente combinadas visando à tarefa a ser executada.

2. Autonomia e adaptabilidade – o robô deve executar suas ações e refinar (ou modificar) a tarefa e seu próprio comportamento de acordo com suas ações e o contexto percebido.

3. Reatividade – o robô deve considerar eventos com limites de tempo compatíveis com formas corretas e eficientes de se atingir suas metas (incluindo sua operação).

4. Comportamento consistente – as reações dos robôs na ocorrência de eventos devem ser guiadas pelos objetivos de sua tarefa.

5. Robustez – a arquitetura de controle deve ser capaz de explorar a redundância das funções de processamento.

6. Extensível – integrar novas funções e definições de novas tarefas deve ser uma tarefa fácil. A capacidade de aprendizado é um fator importante a ser considerado. De acordo com os autores, "a arquitetura deve aprender sempre que possível".

Ao analisar estas características na ótica dos sistemas auto-organizáveis (Ünsal, 1993), nota-se que parte dos desafios propostos por Alami *et al.* (1998) enfocam a capacidade de entender o meio ("Affect") e planejar suas ações sobre este ("Telos"). Ambos os autores notam que a atuação em ambientes não-ideais de sistemas inteligentes – robóticos ou não – dependem não só da programação inicial do sistema, mas também de sua adaptabilidade ao contexto aonde o robô está inserido.

Os pesquisadores vêm buscando arquiteturas de sistemas que permitam esta adaptabilidade. Na Figura 2.3, a arquitetura proposta por Alami *et al.* (1998) é apresentada. Esta arquitetura possui cinco níveis:

- Nível físico: onde é realizada a conversão entre os sinais digitais e analógicos através de sensores e atuadores;
- Nível lógico: realiza o interfaceamento entre o nível físico e o nível funcional;
- Nível funcional: encapsula uma gama de ações elementares do robô, tais como localização, controle do movimento, de garras/braços mecânicos, mecanismos de movimentação de câmera, etc;
- Nível de controle: controla e coordena a execução de funções distribuindo-as de acordo com os requisitos da tarefa;
- Nível de decisão: engloba a capacidade de planejamento de tarefas e supervisionamento de suas execuções. É reativo a eventos do nível anterior. Este nível pode ser decomposto no sistema de planejamento da missão e no sistema supervisório da execução da mesma.

No caso do robô móvel aqui proposto, tem-se os níveis físico e lógico como sendo os projetos mecânico e de interface do ROV. Esta interface permite ao nível funcional, por exemplo, manter a estabilidade de robôs submarinos ou ajustar a intensidade de uma lâmpada para a obtenção de uma melhor imagem de um problema em um duto por um robô de inspeção de tubulações. O nível de controle executa os comandos do supervisor de tarefas, enquanto este executa as tarefas planejadas pelo supervisor de planejamento.

Outro conceito proposto na literatura é o de “arquitetura de robôs reativos”, definido por Ranganatha e Koeing (2003) como sendo uma arquitetura cujo objetivo é a concepção de um sistema rápido baseado apenas nas leituras dos sensores, não possuindo um planejamento a longo prazo. Para o trabalho com robôs de inspeção, um planejamento a longo prazo se faz necessário, pois é através deste planejamento que se busca uma redução de tempo na execução do serviço.

Tal esforço de controle é particionado por Sandi *et al.* (1998) em dois subproblemas principais. Primeiro, o problema de navegação que “designa a determinação de posição e orientação do veículo em um dado instante de tempo” e, segundo, a guiagem, referente ao controle da trajetória do sistema móvel. Na arquitetura proposta por Alami *et al.* (1998),

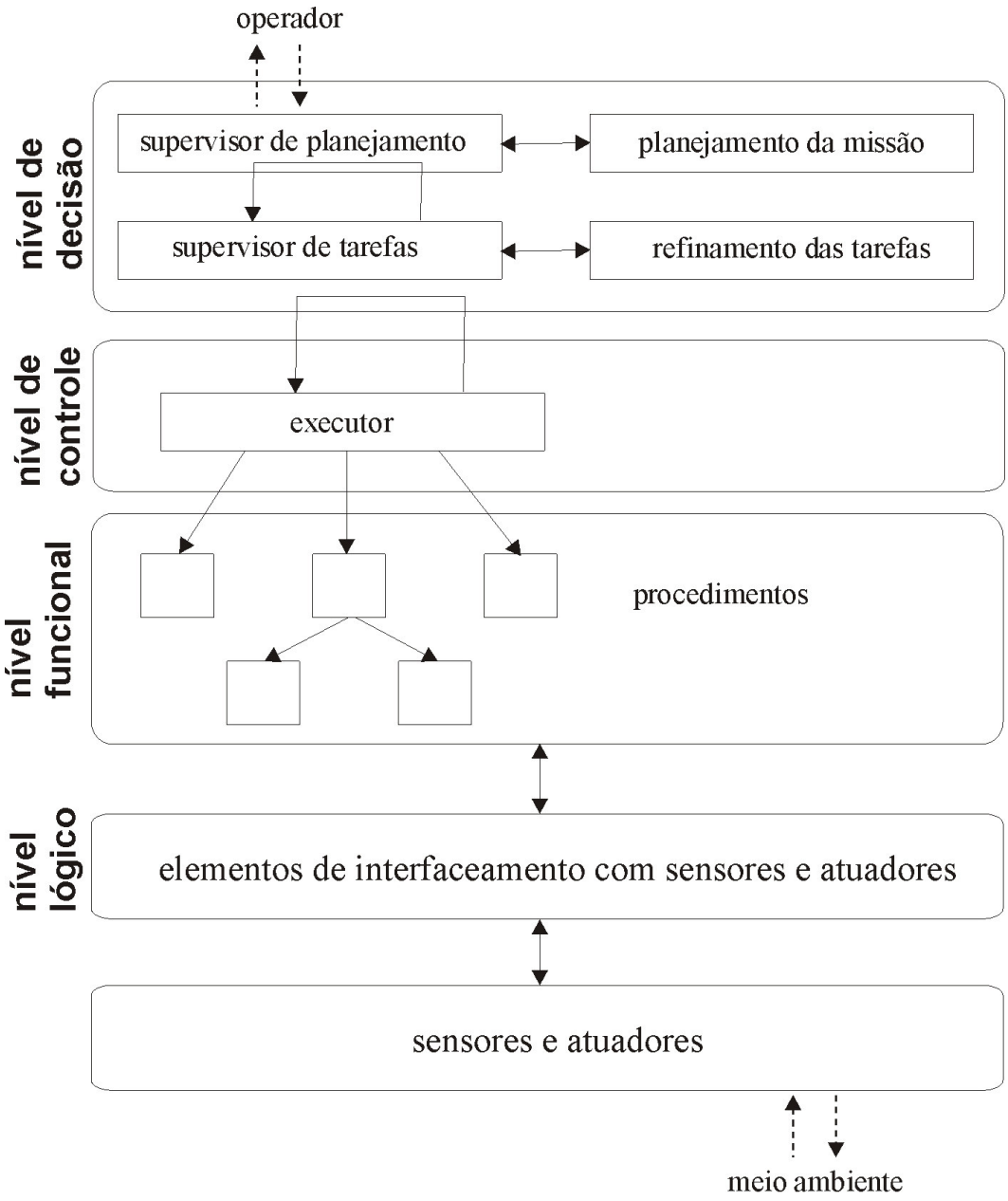


Figura 2.3 Arquitetura de sistemas robóticos adaptativos proposta por Alami *et al.* (1998).

quanto maior o nível hierárquico, maior a preocupação com a navegabilidade do robô. Quanto menor o nível hierárquico, maior a preocupação com a guiagem.

Sandi *et al.* (1998) citam como método clássico de navegação o *dead-reckoning*, método que, segundo os autores, não é imune ao acumulamento de erros no posicionamento do sistema devido às limitações do sensor. Esta técnica, segundo os autores, requer o uso em conjunto de vários outros sensores de forma que se obtenham medidas redundantes da

mesma grandeza e, assim, consiga-se compensar este erro. Sandi *et al.* (1998) citam vários experimentos no qual *encoders* (codificadores digitais de posição) são utilizados junto a acelerômetros e giroscópios para a obtenção da posição, e outros onde informações dos *encoders* são contrabalançadas por um sistema de processamento de imagem.

Segundo Sandi *et al.* (1998), tanto a navegação quanto a guiagem necessitam de uma referência em comum: pontos com posição e orientação desejadas para o veículo (*waypoints*). A obtenção destes pontos é alvo de vários trabalhos. Dixon *et al.* (2002) utilizam uma análise estatística para determinar os objetivos a serem atingidos por um braço robótico, mostrando que esta necessidade não está presente apenas em robôs móveis.

A dificuldade de um projeto sensível à realidade de onde o robô opera foi notada por Hagras *et al.* (2001): "Outro grande desafio na robótica autônoma é a incerteza que caracteriza o ambiente do mundo real. Isso explica porque é difícil, ou realmente impossível, produzir simulações de robôs de operação em meios externos que sejam suficientemente condizentes com a real dinâmica para o projeto e otimização de controladores bons o suficiente." Esta incerteza é maior de acordo com o grau de aproximação do mundo real. Referenciando-se à estrutura da Figura 2.3, nota-se que o tempo de resposta do sistema deve ser menor quanto mais baixo for o nível abordado. As incertezas mencionadas por Hagras *et al.* (2001) são fundamentadas no nível físico, sendo que cada nível superior, ao abstrair os dados, possuem também a tarefa de filtrar informações, só passando os dados pertinentes ao nível superior. Assim, o nível de controle não recebe as variações tênues das leituras dos sensores, assim como o sistema de planejamento da missão não recebe o estado das tarefas, pois é uma tarefa do sistema supervisor de tarefas.

É interessante notar que a simulação da operação do robô em um meio dinâmico como no caso dos ambientes encontrados em inspeções de tubos é complexa. Existem forças referentes a dinâmicas de fluidos (é raro encontrar tubulações sem carga), a diferenças na resposta dinâmica do sistema (devido, por exemplo a trechos escorregadios do percurso formados por acúmulo de limo etc) e a outros problemas. Uma abstração semelhante se faz necessária em outras arquiteturas de controle com planejamento de robôs como a definida por Low *et al.* (2002).

Todas as análises mostradas até o momento, sejam abordagens práticas ou teóricas de automação, mostram um ponto comum: em algum momento se faz necessário o

planejamento de elementos referenciados como *waypoints* (Sandi *et al.*, 1998; Dixon *et al.*, 2002) ou plano de missão (Alami *et al.*, 1998).

Esta dissertação enfoca obtenção automática deste planejamento. As preliminares à topologia do meio e à lista de pontos por onde o robô deve passar para realizar a inspeção devem ser determinadas *a priori*. Cientes de que este meio pode mudar devido a passagens interrompidas, nosso sistema deve ser capaz de, com alterações mínimas, recalcular estes "*waypoints*" no momento em que perceber quaisquer variações.

2.4. Premissas do problema encontradas até agora

As premissas do problema a serem consideradas são as seguintes:

Premissa 1: Deseja-se planejar a rota a ser usada em uma inspeção de um ROV, minimizando o caminho necessário para passar por um conjunto de locações determinadas. Para tal planejamento, pode-se considerar que se possui o mapeamento do local a ser inspecionado.

Premissa 2: Este mapeamento pode ser alterado no decorrer da inspeção (caminhos podem estar obstruídos). Assim, se faz necessário realizar um replanejamento com baixo custo computacional.

Premissa 3: É levada em conta a limitação do cabo umbilical. Assim, mais uma premissa a ser considerada é que existe uma limitação de distância entre o ROV e a base.

No capítulo 3, uma analogia deste problema com o problema do caixeiro-viajante, e uma forma de resolução deste é discutida.

Capítulo 3. Introdução à robótica móvel

Jones *et al*. (1998), caracterizam um robô como sendo "a conexão inteligente entre percepção e ação". Visando a implementação desta conexão, Pfeifer e Scheier (1994) citam a metáfora do processamento de informações, também denominada de metáfora do computador, como algo que obtém uma entrada, realiza um processamento e uma saída é obtida. Quando esta metáfora é aplicada na área de agentes inteligentes, tem-se o ciclo sentir-e-atuar.

Utilizando esta abordagem, as arquiteturas de agentes inteligentes são subdivididos em vários subsistemas: perceptual, de processamento central, motor e variações. Porém, de acordo com Pfeifer e Scheider (1994), estas classificações possuem uma premissa em comum: a hipótese de que eles podem ser divididos e estudados em subsistemas. Os autores exemplificam esta hipótese da seguinte forma:

"*Primeiro, o sistema de percepção é acionado, por exemplo, um copo de cerveja é reconhecido. Depois disso, um sistema de controle central toma o lugar do sistema de percepção, atuando por exemplo como um sistema de tomada de decisão (se o agente está com sede, ele ou ela pode planejar a atividade de beber). E finalmente, o plano é transferido para uma sequência de ações motoras (levantando o copo, inclinando-o, etc). Em outras palavras existe uma cadeia de eventos ligando a percepção à ação*" (Pfeifer e Scheider, 1994).

Os autores criticam esta abordagem, defendendo que a sensação não pode ser processada de uma forma tão analítica quanto conclui o senso comum. A conexão inteligente entra a percepção e a ação se dá não através da análise da percepção, mas sim da

busca de uma percepção para que se consiga uma ação determinada. Citando Pfeifer e Scheider (1994):

"*O organismo está sempre envolvido em uma determinada ação ou atividade e esta determina o tipo de sensoriamento a ser realizado. Mais que isso, determina a interpretação deste estímulo. Esta interpretação será muito diferente quando se olha para uma bebida ou quando se escreve um artigo científico. Assim, talvez ir da percepção à ação não seja a direção correta, sendo preferível, ao invés disso, que se inicie pela ação.*"

Marchi (2001) cita três categorias principais aonde os robôs atuais podem ser enquadrados: robôs industriais, de serviço e de campo.

Marchi (2001) define robôs industriais como "em geral, plataformas móveis utilizadas para tarefas pesadas como o transporte de materiais e produtos finais em sistemas de manufatura." Ainda de acordo com a autora, estes robôs são denominados AGVs – *Automated Guied Vehicles* – e são concebidos para atuar em ambientes altamente estruturados seguindo marcações no piso.

Como robôs de serviço, são classificados aqueles utilizados para limpeza em geral (de pisos, dutos de ar etc), vigilância e no transporte de materiais leves. O ambiente neste caso é estruturado, e o robô possui um conhecimento prévio do mesmo (Marchi, 2001).

Os robôs de campo, trabalham "em ambientes não estruturados, pouco conhecidos e geralmente perigosos" (Marchi, 2001). São utilizados para a exploração (espacial, espeleológica, etc), mineração, limpeza de acidentes nucleares, navegação em estradas e em tarefas agrícolas. Fuller (1999), citam alguns destes robôs:

"O espaço sideral é um meio hostil para os seres humanos, então robôs são usados para muita da exploração espacial. *Viking 2* pousou em Marte, a *Voyager* investigou Saturno e hoje deixou nosso sistema solar e se encontra em um local desconhecido; *Galileo*

voou por Júpiter e o *Mars Rover* explorou Marte. Ao contrário à sonda *Viking*, o Mars Rover realizou grande parte de suas próprias decisões, em vez de depender de comandos humanos provenientes da Terra.

A exploração de grandes profundidades oceânicas é perigosa para seres humanos. Pessoas já vem usando manipuladores controlados remotamente para auxiliar esta exploração. Como o uso de manipuladores controlados remotamente possui um valor limitado neste meio, os pesquisadores tentam desenvolver robôs autônomos para o uso exploração de ambientes oceânicos. No presente momento, companhias que mantém tubulações se utilizam de robôs para realizar a inspeção por dentro dos tubos. Uma versão modificada deste robô pode ser usada para inspecionar dutos externamente quando estes forem submersos."

Costa (1998), sintetiza duas formas de descrição de um sistema robótico autônomo: uma primeira, trata dos diferentes módulos do sistema enquanto outra trata da criação de soluções baseadas em comportamento.

Quando se trata o sistema por módulos, o resultado é um conjunto de módulos funcionais interdependentes distribuídos entre camadas de sensores e atuadores. Neste caso, a definição de ações a serem realizadas ficam em geral sob controle de um módulo de planejamento. (Costa, 1998).

As soluções baseadas em comportamentos são baseados na construção de um sistema robótico composto de módulos independentes, definidos por Costa (1998) como "subsistemas capazes de operar sozinhos e realizar tarefas específicas". Tais sistemas, que executam as tarefas através de recursos limitados de informação e atuação, são mais fáceis de implementar e possuem um custo computacional menor (Costa, 1998). A interligação

dos subsistemas é realizada de forma hierárquica, dividindo os módulos por níveis de comportamento.

Como comportamento, Costa (1998) define como “uma ação motora objetiva ligada a um processo específico de percepção”. Se aprofundando mais, o autor ainda diz que “um comportamento é definido como um sistema que resolve um problema específico, gerindo a informação proveniente de alguns sensores e controlando alguns atuadores, utilizando para isso recursos computacionais limitados, interagindo, normalmente, de uma forma reativa com o meio ambiente”.

Marchi (2001) define a construção de sistemas robóticos como “uma tarefa multidisciplinar, envolvendo elementos das engenharias elétrica, mecânica e de ciência da computação”. Ao se analisar trabalhos como o de Zimmer (1995), notamos que, dependendo da abordagem utilizada no desenvolvimento do sistema autônomo, outras áreas podem ser incluídas, como psicólogos, fisiologistas, neuro-cientistas, entre outros.

Zimmer (1995) analisa três tarefas importantes que o robô autônomo deve ser capaz de realizar:

- obter sua localização, mesmo estando em um ambiente desconhecido e não estruturado;
- realizar navegação, englobando duas tarefas, a de “navegação global”, que se refere ao planejamento de trajetórias e a “navegação local”, que cuida da realização de manobras para o cumprimento da trajetória;
- determinar um alvo, quando se requer do robô ações que vão além da navegação pura e simples (como por exemplo transporte de cargas, vigilância etc).

Marchi (2001) sintetiza que a principal dificuldade em se construir um sistema de navegação (“local”) de um robô móvel autônomo é permitir ao robô trafegar entre dois pontos sem que haja colisão com obstáculos. As principais dificuldades encontradas nesta tarefa, ainda segundo a autora, são aquelas relacionadas ao mundo real, que possui características que não podem ser desprezadas no projeto de um sistema autônomo:

- o mundo real não é totalmente acessível, pois os sensores oferecem medições imprecisas e algumas mudanças do meio são percebidas apenas quando o robô está fisicamente próximo a elas;
- é indeterminado, pois sempre rodas podem escorregar, baterias podem falhar, sendo assim impossível assegurar que uma determinada ação seja executada sem falhas;
- não é totalmente previsível, pois suas ações futuras e efeitos não podem ser pré-determinados com exatidão;
- dinâmico, sendo necessário o robô autônomo entender quando é melhor esperar e quando é melhor agir imediatamente;
- contínuo, sendo que as ações e estados ocorrem de acordo com as mudanças no próprio meio. Como existem infinitas combinações, é impossível enumerar o conjunto de ações.

Tais características, podem também ser um problema para o sistema de navegação global (ou planejamento de caminhos), pois podem causar passagens interrompidas, dificultar trecho e facilitar outros, fornecendo assim novas características ao meio previamente conhecido.

Assim, para que um robô móvel possa atuar no mundo real, ele deve ser capaz de se adaptar às suas mudanças. Conforme define Marchi (2001), a estrutura do sistema autônomo deve ser híbrida, consistindo de duas partes principais:

- Uma primeira, que trata o ambiente como um problema estático, e realiza o planejamento de rotas a partir deste conhecimento. Esta abordagem, aonde temos um modelo de ambiente fornecido *a priori*, é chamada de Abordagem Planejada ou Deliberativa.
- Uma segunda, que garante robustez e adaptabilidade ao robô, aonde o mesmo assume comportamentos (como desviar de obstáculos, seguir paredes, caminhas em uma direção, etc) e toma suas decizões baseando-se na leitura de seus sensores.
- Com isso, consegue-se navegar por ambientes dinâmicos e não estruturados.

Na abordagem híbrida, "o robô conhece o ambiente, planeja a trajetória e age reativamente diante de situações imprevistas" (Marchi, 2001). Ou seja, existe um plano inicial de trajeto, e um sistema de navegação pronto para responder às respostas inesperadas do meio. Tal habilidade é citada por Nehmzow (2001), que afirma que antes que se tente fazer um robô móvel desempenhar uma tarefa complexa, é essencial que o robô opere de maneira segura, realizando comportamentos simples como evitando obstáculos e pessoas.

O problema de planejamento de caminhos pode ser definido como uma variação do planejamento de ações de um outro sistema robótico: robôs manipuladores. Siegwart e Nourbakhsh (2004, pp.259), afirmam que "o problema de planejamento de trajetórias para um robô manipulador com, por exemplo, seis graus de liberdade, é mais complexo que um robô móvel de tração diferencial operando em um ambiente bidimensional. Assim, embora possamos obter inspiração de técnicas inventadas visando robôs manipuladores, algoritmos de planejamento de trajetórias usados por robôs móveis tendem a possuir simplificações

graças ao número reduzido de graus de liberdade do sistema. Mais ainda, os robôs manipuladores industriais normalmente buscam operar na maior velocidade possível, levando em conta o impacto econômico envolvido. Assim, deve-se levar em conta a dinâmica, não apenas a cinemática de seus movimentos. Em contraste, o número de robôs móveis que operam em baixas velocidades é grande, o que faz estas considerações serem raramente consideradas durante o planejamento de trajetórias, simplificando ainda mais a instanciação do problema."

Este planejamento de tarefas complexas, segundo Choset *et al.* (1996), pode ser feito de duas formas distintas:

- Através do método denominado pelo autor de "planejamento clássico" (*classical planning*), que requer um conhecimento antecipado e total do contexto aonde o robô está inserido
- Ou através do método de "planejamento baseado em sensores" (*sensor based planning*), que, em contraste ao planejamento clássico, integra os sistemas de sensoriamento do robô com o processo de planejamento. Choset *et al.* (1996) e Choset e Nagatami (2001), adicionam o conceito de *roadmap n*a aplicação deste método de planejamento, visando possibilitar ao sistema de planejamento três características: acessibilidade (achando um caminho livre pelo roadmap), conectividade (permitindo ao robô se mover em direção à vizinhança do objetivo) e ainda partição do planejamento (construindo um caminho livre de colisões entre os vários pontos que ligam o local de início do movimento até o objetivo).

Todas as abordagens, porém, levam em conta uma variedade limitada de elementos sensores. McKerrow (1991), define os usos principais dos sensores em robôs autônomos:

- medir os parâmetros do próprio robô para laços de controle (por exemplo, um encoder pode fornecer a posição angular de uma roda, permitindo assim aumentar ou diminuir o torque do motor correspondente);
- determinar a localização de objetos (por exemplo, um alvo a ser seguido);
- correção de erros no modelo matemático do robô ou do mundo (por exemplo: erro de sensores, mudança no contexto do robô, etc);
- detectar e evitar situações de falha (como por exemplo evitar um buraco no solo);
- detectar e evitar colisões (seja contra objetos estáticos ou mesmo objetos móveis);
- monitorar interações com o meio (por exemplo, a passagem por uma mancha de óleo);
- monitorar mudanças no meio que possam vir a atrapalhar a tarefa (mudanças de temperatura, por exemplo);
- inspecionar os resultados dos processos (para garantir que uma tarefa foi realmente realizada).

Para tal, os elementos sensores devem ser capazes de transformar uma grandeza física em uma percepção válida para o robô. McKerrow (1991) define o processo de sensoriamento de um robô da seguinte forma:

- uma **grandeza física** é transformada em um **sinal elétrico** por um transdutor;
- este **sinal elétrico** é condicionado por uma eletrônica de condicionamento, se tornando um **sinal condicionado**, normalmente após etapas de linearização e amplificação;

- o **sinal condicionado** passa por uma interface computacional, que o transforma em um **sinal digital**;
- este **sinal digital** é processado, gerando um **modelo geométrico** da grandeza que se deseja perceber, sendo que este pode conter informações de diversos sensores (como uma matriz de contato, ou até mesmo um sinal multiplexado no tempo como é o caso de um sinal de vídeo);
- por fim, esse **modelo geométrico** é analisado com o intuito de se conseguir determinar o estado atual do mundo e se inferir o resultado de possíveis ações do sistema robótico. O resultado desta análise é denominado por McKerrow (1991) como sendo a "**percepção**".

Gaspar *et al.* (2004) analisa que a forma de percepção do mundo por robôs como uma ação cujo resultado depende do contexto aonde o sistema autônomo está inserido. O autor exemplifica, dizendo que "quando andamos em uma avenida, é suficiente que se conheça a posição atual com precisão de uma quadra. Porém, quando se deseja entrar passar por uma porta, a precisão requerida é muito maior. Este raciocínio é fundamentado quando observa-se que, na natureza, muitos animais se utilizam de navegação baseada em pontos específicos do terreno (*landmarks*) e métodos de integração de rotas."

Também defendendo que a percepção não pode ser tratada de forma separada à ação, Wasson *et al.* (1999) afirma que "a percepção precisa conhecer a ação e a ação precisa conhecer a percepção. Ou seja, um sistema que perceba o mundo deve poder ser direcionado e controlado de acordo com a tarefa atual do agente. A ação, por sua vez, precisa de conhecer a natureza da informação disponibilizada pelo sistema de percepção".

Esta conexão, notada claramente em arquiteturas robóticas reativas, pode ser combinada com elementos de processamento que geram padrões comportamentais esperados que dificilmente seriam gerados de forma puramente reativa. Estas arquiteturas híbridas, combinam formas diferentes de mecanismos de planejamento com níveis "mais baixos" de controles baseados em comportamento. As arquiteturas híbridas consistem hoje no maior foco no desenvolvimento de agentes autônomos e semi-autônomos operando em tempo real e ambientes reais (Datteri *et al*., 2003).

Tanto em arquiteturas híbridas quanto em arquiteturas reativas, temos um conjunto limitado de grandezas que o robô pode medir. A seguir, é mostrado um conjunto de sensores e atuadores encontrados comercialmente com frequência.

3.1. Sensores de uso comum na robótica móvel

3.1.1. Sensores de proximidade

Para evitar colisões, são encontrados no mercado módulos detectores de obtáculos baseados em infravermelho (Figura 3.1). Tais módulos, possuem um emissor de luz infravermelha e um receptor, e se baseia na premissa de que é possível detectar um obstáculo quando este estiver próximo do robô o bastante para que o feixe de infravermelho seja detectado pelo receptor.

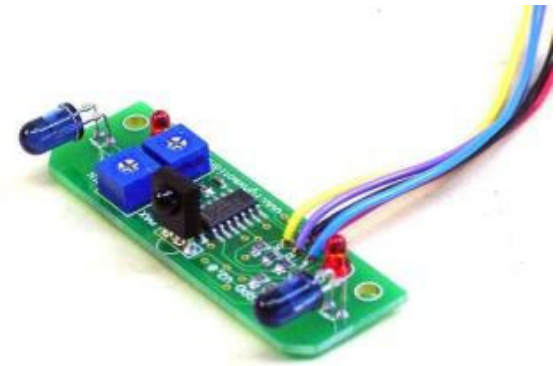


Figura 3.1 Sensor de proximidade utilizando um conjunto emissor/detector de infravermelho (fonte: LYNXMOTION, 2005).

Para detectar a proximidade de obstáculos, também se é utilizado sensores de ultrassom, como os existentes no robô Nomad 200 (Queiroz *et al*., 1996). Estes sensores, permitem que o sistema móvel obtenha a distância entre o robô e o objeto em que o sensor está focalizado. O robô Nomad 200 é mostrado na Figura 3.2.

Figura 3.2 O robô Nomad 200 (fonte: Queroz *et al*., 1996).

Outro tipo de sensor comum é o sensor de toque mecânico (Figura 3.3). Este sensor é popular principalmente pela sua simplicidade no tratamento do sinal (pois consiste apenas de um interruptor que chaveia entre dois níveis lógicos).

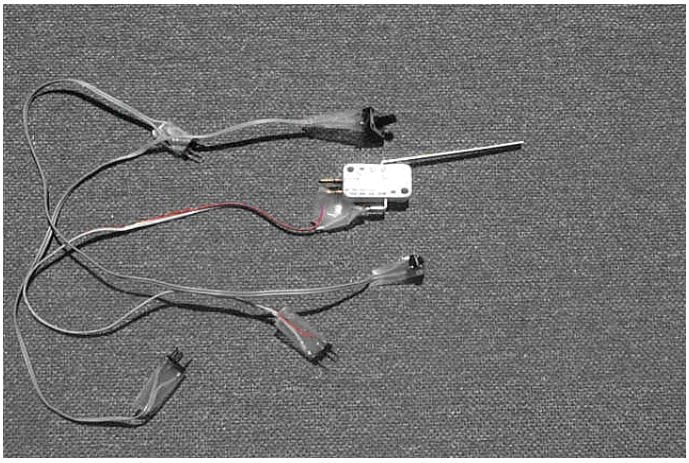


Figura 3.3 Um exemplo e sensor de toque (fonte: Instituto para a Robótica Prática, 2005).

3.1.2. Sensores de deslocamento

Utilizados para inferir a posição atual dos robôs, os *encoders* são um tipo de sensor de deslocamento muito encontrado no mercado. Sua importância é tamanha que empresas fornecedoras de motores de precisão já incluem *encoders* acoplados aos seus motores.

Os *encoders* consistem basicamente de um disco com aberturas ("janelas") estrategicamente localizadas acoplado à um sensor ótico transmissivo. Desta forma, contando o número de vezes o sensor é chaveado, consegue-se saber o número de janelas passaram, e, desta forma, consegue-se saber quanto um determinado eixo girou.

É interessante notar que os encoders estão sujeitos ao deslizamento das rodas. Desta forma, embora sejam uma boa fonte de estimativa de deslocamento, o "ajuste fino" da posição do robô deve ser realizada através de outro estímulo.

De interpretação um pouco mais complexa, os acelerômetros também podem ser usados para a determinação do deslocamento, assim com a velocidade instantânea do robô. Na Figura 3.4 tem-se exemplos de acelerômetros disponíveis no mercado. Porém, devido a alguns fatores, tais como custo (muito maior que o *encoder*) e custo computacional envolvido para sua análise (pois a determinação do espaço percorrido é a integral no ponto da aceleração), estes sensores não são explorados em vários robôs comerciais, como o Nomad 200 e o Kephera.

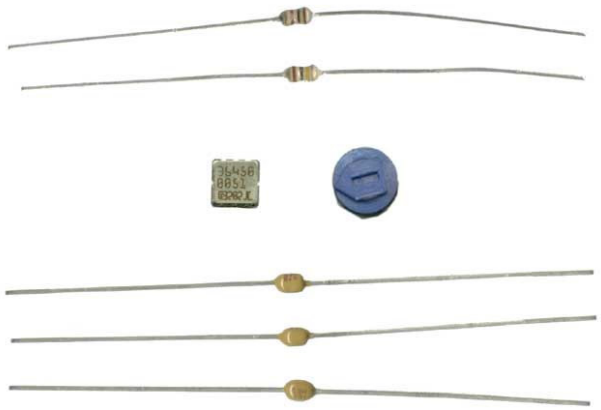


Figura 3.4 Exemplos de encapsulamentos de acelerômetros disponíveis no mercado (fonte: P.A.R.T.S., 2005).

3.1.3. Câmeras

Hoje, nota-se em diversos robôs de laboratório o uso de câmeras. Estes sensores, embora venham com a dificuldade do tratamento de vídeo em tempo real, permitem a extração de informações importantes do meio, por exemplo, a localização de obstáculos, localização de objetivos etc.

3.1.4. Outros sensores e a ambiguidade sensorial

Comercialmente e em desenvolvimento, encontra-se no mercado uma vasta gama de sensores que também podem ser utilizados na robótica móvel. Entre eles, temos bússulas eletrônicas (Zagros Robotics, 2005) entre outros.

A lista mostrada até agora não tem o intuito de esgotar as possibilidades de grandezas e formas de medição que possam ser percebidas por um sistema robótico móvel. Muito antes pelo contrário, o número de universidades e empresas que pesquisam novas formas de sensoriamento é vasto, como por exemplo a empresa Sensor Research (Sensor Research 2005). O que foi apresentado representa apenas os sensores mais comuns.

O projetista de sistemas robóticos também não pode ignorar a conveniência do uso combinado de vários tipos de sensores que objetivem a medida de uma mesma grandeza. Segundo Monteiro e Ribeiro (2004), a ambiguidade sensorial se torna mais interessante principalmente quando se percebe que nenhum sensor dá uma informação completa do meio. Segundo os autores, "frequentemente, o estado não é totalmente observável, ou seja, não é sempre conhecido qual o estado atual em que o sistema está. Nas situações práticas, os sensores do agente fornecem apenas informações parciais (e com freqüência ruidosas), que não conseguem perceber o estado real do ambiente". Neste caso, Monteiro e Ribeiro (2004) se utilizaram de uma arquitetura baseada em redes neurais para realizar a integração das leituras de sensores como sonar, laser e infravermelho.

São observados outras arquiteturas que possibilitam a integração entre sensores, como o trabalho de Sandi *et al.* (1998), que integra odômetros e bússula digital. Neste trabalho, os autores escolheram uma abordagem baseada em filtros de Kalman para a obtenção da posição do robô móvel.

Dando continuidade à esta introdução à robótica móvel, será apresentada a seguir as morfologias mais comuns de robôs móveis encontrados comercialmente.

3.2. Morfologias de robôs disponíveis comercialmente

3.2.1. Robôs móveis com rodas

Uma arquitetura comum quando tratamos do robôs móveis, é a arquitetura de robôs dotados de rodas (figura 3.5). Existem váriações desta arquitetura, sendo a mais divulgada ao público a arquitetura de seis rodas com amortecedores independentes, existente no robô Sojourner enviada pela NASA ao planeta Marte (figura 3.6).

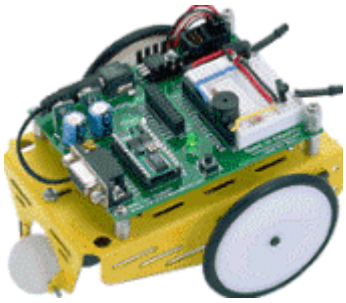


Figura 3.5 Robô com duas rodas de acionamento independente (fonte: Robot Books, 2005).

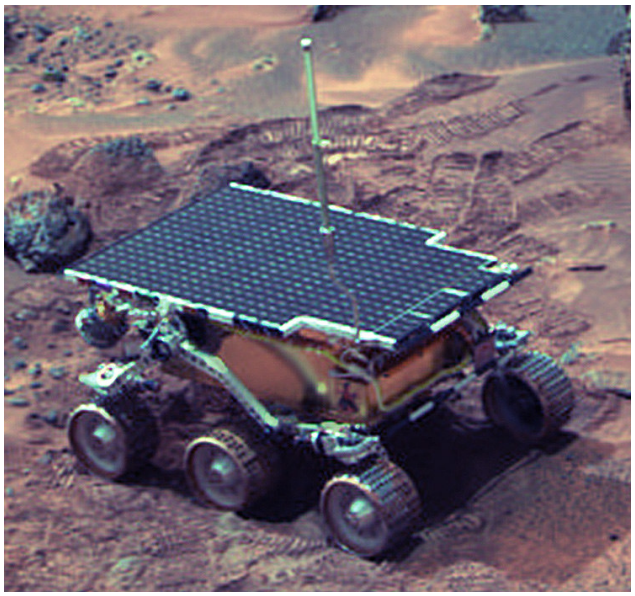


Figura 3.6 O robô Soujourner da NASA (fonte: NASA, 2005).

Outras arquiteturas, como a mostrada por Choi e Ryew (2002), se utilizam de sistemas de rodas articuladas que, juntamente com um corpo segmentado, permitem uma grande manobrabilidade dentro de tubos de gás.

3.2.2. Esteiras

Utilizadas quando se busca um atrito maior do solo, esteiras como as mostradas na Figura 3.7.

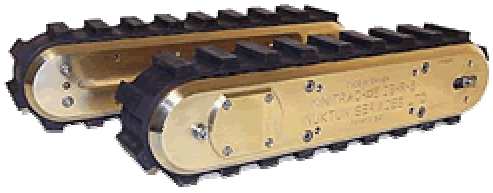


Figura 3.7 Conjunto de esteiras prontas para serem usadas em um robô de inspeção (fonte: Inuktum, 2005).

3.2.3. Pernas

O simples ato de andar traz à tona uma série de problemas, a maioria relacionados à manutenção do equilíbrio. Apenas resolver este problema mecatrônico é um dos campos de destaque da pesquisa em robótica atual.

Robôs com quatro pernas já são realidade no mercado, como o cão-robô Aibo da Sony (Figura 3.8). Porém, o desafio de se criar um robô bípede é maior. Este desafio, vêm sido colocado à comunidade científica, como em competições como as promovidas pela FIRA (Figura 3.9).

Figura 3.8 O robô cão AIBO da Sony (fonte: Sony, 2005).

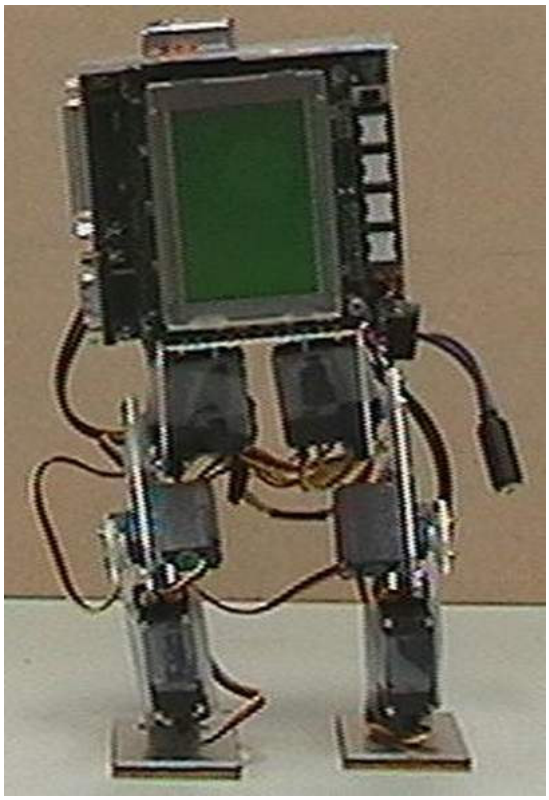


Figura 3.9 Um dos robôs bípedes apresentados em eventos da FIRA (fonte: FIRA, 2005).

Capítulo 4. Metodologia

4.1. Problema do caixeiro viajante

As origens do problema do caixeiro viajante (TSP – *Travelling Salesman Problem*) são obscuras, de acordo com Applegate *et al.* (1998). Os autores enumeram trabalhos relevantes no estudo do algoritmo: a primeira referência citada pelo autor é uma publicação do matemático e economista Karl Menger na década de 20. Na década de 30, o problema reapareceu em grupos de estudo da Universidade de Princeton. Na década de 40, foi estudado por estatísticos que ligaram este problema com aplicações na agricultura, e o matemático Merril Flood popularizou o problema entre seus colegas da corporação RAND na Califórnia.

Ravikumar *et al.* (1998) define o problema do TSP como:

"O problema do caixeiro viajante é um problema bem conhecido de otimização NP-difícil. Pode ser definido da seguinte forma. Seja $G = (V, E)$ um grafo, onde $V = (v_1...v_n)$ representa o conjunto de vértices e $E = ((v_i, v_j): v_i, v_j \in V)$ represente o conjunto de arestas. A cada aresta (v_i, v_j), $(i \neq j)$, é associado um custo não negativo c_{ij} . O TSP consiste em encontrar uma rota fechada de custo mínimo passando por todos os vértices apenas uma vez."

Helsgaun (1998) classifica o problema do TSP em três categorias, a partir da análise da matriz de custos que possui todos os elementos c_{ij}:

- se $c_{ij} = c_{ji}$, o problema é dito simétrico. Caso contrário, é dito assimétrico;
- se a inequação $c_{ik} \leq c_{ij} + c_{jk}$ se mantém para todo i, j, k, o problema é dito métrico;
- se c_{ij} são as distâncias euclidianas entre pontos no plano, o problema é dito Euclidiano. Um problema Euclidiano é, também, simétrico e métrico.

Cunha *et al.* (2002), assim como Ravikumar *et al.* (1998) e Helsgaun (1998), citam o trabalho de outros pesquisadores, demonstrando a vasta gama de aplicações em engenharia, ciência e bioinformática da resolução deste problema. Entre eles, são citados o

uso deste para a soldagem de placas de circuito impresso em fábricas automatizadas, cristalografia de raio-x e controle de robôs. Existem outras heurísticas, como o planejamento de rotas de veículos de entrega, o problema de atribuição quadrática, entre outros discutidos na literatura (Boctor *et al.*, 2003). Para todas as aplicações mencionadas, nota-se a necessidade de heurísticas de baixo custo computacional e rápida convergência, conforme mencionado por Ravikumar *et al.* (1998).

O conceito de problema de otimização combinatória, é definido por Blum (2003). Segundo o autor, $P=(S,f)$ é um problema de otimização combinatório quando possui:

- Um conjunto X de variáveis, tal que $X = \{x_1,...x_n\}$
- Domínios $D_1,...D_n$ das variáveis $x_1,...,x_n$
- Condições de contorno para cada variável
- Uma função objetivo f a ser minimizada onde $f:D_1 x...x D_n \rightarrow R^+$.

O TSP é considerado um problema de otimização combinatória não determinístico e de complexidade polinomial categorizado como NP-difícil (do inglês *NP-Hard, Nondeterministic-Polynomial-Hard, não determinístico-polinomial-difícil*), indicando sua complexidade exponencial. Assim como o caixeiro viajante, outros problemas combinatórios são categorizados como NP-difícil. Entre eles, tem-se (Silver 2002):

- O problema de atribuição quadrática (*quadratic assignment problem*), definido como: dado um conjunto de n depósitos, o volume de material que pode ser movido por unidade de tempo entre dois depósitos é conhecido *a priori*. O objetivo é posicionar os armazéns de forma a minimizar a relação distância *versus* volume de material transportado.
- O problema do planejamento de projeto com recursos limitados (*resource-constraint project-scheduling problem*), definido como: dado um projeto com n atividades distintas, cada uma composta de um tempo de execução e um certo número de recursos. Existe um valor máximo para o uso simultâneo de cada recurso, assim como relações de dependências entre as atividades. O objetivo é minimizar a execução de todas as tarefas (e, consequentemente, do tempo total do projeto).
- O problema de determinação de rotas de veículos de entregas (*vehicle routing problem*), definido como: dado um depósito e um conjunto de destinos,

determinar a melhor forma de distribuir uma frota de veículos para a entrega de modo a respeitar situações de contorno, a saber:

(a) máximo de carga disponível por veículo;
(b) máximo de tempo (ou distância) de viagem por veículo;
(c) situações relacionadas à janela de tempo, como por exemplo horários pré-estabelecidos de liberação de mercadorias ou de entrega de mercadorias.

Cunha *et al.* (2002) mostra a dificuldade de resolução do problema do caixeiro viajante, ao afirmar que "o esforço computacional para a sua resolução cresce exponencialmente com o tamanho do problema". Stager (2001) quantifica esta afirmação ao dizer: "Uma solução para o problema do caixeiro viajante pode ser calculada comparando cada rota com as demais até que o menor caminho é encontrado. O número de rotas que necessitam ser calculadas é igual a $(N\text{-}1)!$, onde N representa o número de cidades. Se um computador pudesse calcular a distância de cada rota em 1ms, a solução para 10 cidades poderia ser encontrada em 3,63 s. Para 15 cidades, a solução seria encontrada em 15 dias e para 20 cidades a solução seria encontrada em 77146 anos" (Stager, 2001) .

O problema apresentado no Capítulo 2 (que é o problema de trajetória considerando as limitações físicas do ROV), mostra similaridades ao problema do TSP, pois:

- ambos os problemas tratam de planejamento de rotas;
- no TSP o objetivo é passar por todas as cidades conhecidas, e o ROV de inspeção deve passar por locais pré-determinados. Em um pior caso, o número de locais a serem inspecionados será igual ao número de locais conhecidos.

São também adicionadas algumas novas condições de contorno ao problema do TSP:

- a limitação do cabo umbilical;
- a necessidade de recalcular a rota em tempo hábil caso o robô perceba uma mudança no meio.

A seguir, algumas formas de resolver o problema do TSP são avaliadas na esperança que, com a adição de algumas heurísticas, a transformação deste problema clássico no problema discutido até o momento seja apropriada.

4.2. Algoritmos para o problema do caixeiro viajante

Helsgaun (1998) classifica as respostas possíveis do problema do caixeiro viajante em duas classes principais:

- algoritmos exatos;
- algoritmos aproximados (ou heurísticos).

Os algoritmos exatos possuem a característica de, após um número determinado de passos, obter a resposta ótima para o problema. Hoje, é possível obter a resposta através destes algoritmos para formulações do TSP com poucas centenas de cidades, embora existam relatos de soluções com milhares de cidades.

Helsgaun (1998) cita os algoritmos de plano de corte e descobrimento de facetas como os mais precisos. Porém, também mostra que são algoritmos complexos, de alto poder computacional ("levou 3-4 anos aproximadamente de tempo de CPU em uma complexa rede de computadores para se determinar a solução exata do problema de 7393 cidades" (Helsgaun, 1998)).

Devido a sua complexidade, foi considerado um dos marcos no estudo deste algoritmo o trabalho de Dantzig, Fulkerson e Johnson em 1954, que publicaram o método de plano de corte que resolve uma instância do TSP de até 49 cidades (Applegate *et al.*, 1998).

Em contraste com os algoritmos exatos, os algoritmos heurísticos visam obter uma resposta aceitável em um tempo menor. Esta resposta, nem sempre é a ótima, porém o ganho de tempo de CPU normalmente compensa este diferença de "apenas alguns pontos percentuais na média".

Silver (2002) analisa três circunstâncias aonde o uso de um método heurístico é vantajoso:

- quando um grande aumento no número de elementos é observado no domínio da função;

- quando existe dificuldade em analisar a função objetivo, muitas vezes pela presença de variáveis estocásticas;
- quando se tem uma função objetivo com comportamento diferenciado em relação ao tempo, necessitando uma análise de séries históricas, não bastando uma única análise de uma única solução em um ponto do espaço de busca.

Silver (2002) cita algumas razões que incentivam o uso de algoritmos heurísticos:

- implementação simples: de acordo com o autor, "a aceitação e uso de regras de decisão pelos tomadores de decisão é mais fácil quando, pelo menos de forma intuitiva, se compreende como as regras operam, em especial os efeitos dos parâmetros do algoritmo na escolha de ações";
- possibilidades fortes de melhorias frente à métodos "tradicionais", quando gerentes podem escolher a modificação de processos consolidados visando o aumento de desempenho do processo produtivo;
- resultados mais rápidos e razoáveis (não necessariamente ótimos), podem ser obtidos através de algoritmos heurísticos;
- robustez, pois os algoritmos heurísticos são menos sensíveis à variações do problema que os algoritmos exatos.

Blum (2003) enumera algumas características de algoritmos baseados em heurísticas. Antes de tudo, o autor define que "heurísticas são estratégias que guiam o processo de busca", com a finalidade de explorar de forma eficiente as soluções possíveis para que se obtenha soluções próximas ao ótimo. As técnicas que constituem algoritmos heurísticos podem variar de simples procedimentos de busca até processos complexos de aprendizado, e são normalmente não determinísticas. Também se enumera que esta classe de algoritmos não é realizada para problemas específicos, embora possa ser previsto a influência sobre condições de contorno do meio aonde está inserido sobre o mesmo.

É interessante notar que Blum (2003) destaca que "as heurísticas avançadas usam o conhecimento da busca incorporado em alguma forma de memória para guiar os passos do agente." Exemplos de conhecimento da busca são mostrados no item 4.3.3 (algoritmo de colônia de formigas) e no Capítulo 5.

Indo mais além, Helsgaun (1998) divide os algoritmos heurísticos em três classes principais:

- algoritmos de construção de caminhos, onde os caminhos são adicionados à rota gradualmente;
- algoritmos de melhoramento de caminhos, onde se têm uma rota e os caminhos são trocados para se obter a melhor rota; e
- algoritmos compostos (ou híbridos), que combinam as duas classes anteriores.

Laporte (2000) apresenta um estudo de heurísticas para a resolução do problema de descoberta de rotas para veículos. Segundo ele, existem três categorias de resposta à este problema:

- heurísticas construtivas, que constroem a solução gradualmente, sempre observando o custo da solução, porém sem obter uma fase exclusiva de otimização;
- heurísticas de duas fases, classificação dos vértices em rotas possíveis e a construção das rotas propriamente ditas;
- métodos de melhoramento, que buscam melhorar cada solução possível ao realizar a troca de rotas nas rotas disponíveis.

Laporte (2000) ainda cita seis algoritmos principais para a resolução desta variação do problema do TSP: *simulated annealing*, *deterministic annealing*, busca tabu, algoritmos genéticos, colônia de formigas e redes neurais artificiais. Outras propostas também são encontradas na literatura, como mapas auto-organizáveis (Vieira, 2003), algoritmos meméticos, *scatter search*, *randomized greedy search* (GRASP), busca em árvore, entre outras (Merz e Freisleben, 2000).

Mills *et al.* (2004) faz a comparação entre os algoritmos heurísticos de minimização, enumerando cinco características principais quanto ao uso de:

- movimentos aleatórios para escapar de mínimos locais;
- soluções baseadas em populações de agentes;
- modificação da vizinhança;
- pesos ou penalidades para a alteração da função objetivo;
- modelos probabilísticos;
- memória, seja ela de longo ou de curto prazo.

4.2.1. GRASP (*Greedy Randomized Adaptive Search Procedure*)

Resende e Velarde (2003) definem o algoritmo GRASP como um método multi-processo, onde cada instância GRASP consiste na construção de uma solução individual aleatória seguida de uma busca local que se utiliza da solução individual como ponto de partida. Este procedimento é realizado repetidamente, sendo que a melhor solução de todos os procedimentos GRASP executados é definida como a melhor solução encontrada.

4.2.2. *Simulated annealing*

Karlson (2002) define o algoritmo de *simulated annealing* através da sua origem física: o processo de *annealing*. Este procedimento, se dá quando um metal aquecido é resfriado. No início, a estrutura cristalina do metal se encontra com um número considerável de imperfeições. Conforme se esfria o metal até a temperatura ambiente, nota-se um rearranjo da estrutura cristalina do mesmo. Visando obter uma estrutura cristalina perfeita, são realizados dois movimentos: um, de aquecimento, que visa buscar o rearranjo aleatório das partículas; outro, de resfriamento, de forma a garantir que se estabeleça a melhor estrutura cristalina.

Ao se simular este procedimento, existem quatro pontos principais a serem tratados no caso (Kirkpatrick *et al.*, 1983):

- determinar a estrutura do elemento;
- determinar sua função de redução de temperatura;
- determinar a forma de rearranjo dos átomos;
- determinar uma função objetivo a ser otimizada.

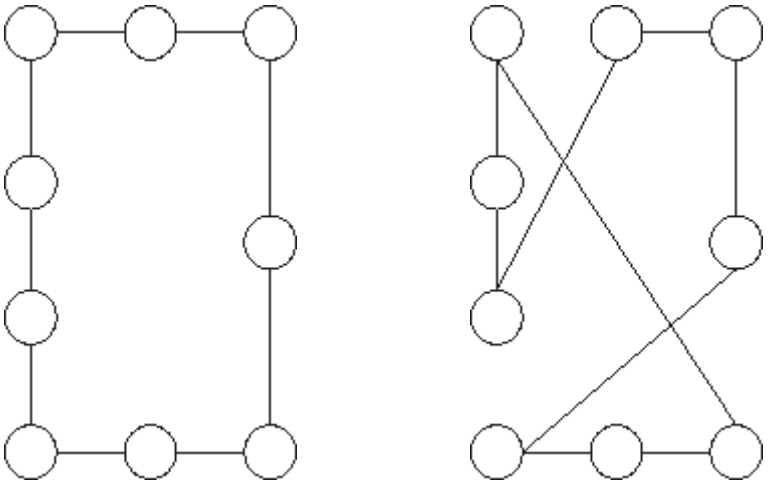
Através deste algoritmo, os pesquisadores têm conseguido resolver diversos problemas combinatórios, entre eles, o problema do caixeiro viajante (Kirkpatrick *et al.*, 1983).

4.2.3. Algoritmo de Lin-Kernighan

Um método clássico na literatura de algoritmos de melhoria de caminhos é o algoritmo 2-opt. Helsgaun (1998) define este como: dado um caminho já fornecido, substitua duas de suas conexões de modo a formar um caminho mais curto. Continue até que não seja possível nenhuma forma de melhoramento. A figura 4.1 mostra um exemplo desta operação com o caminho original (figura 4.1(a)) e com a substituição (figura 4.1(b))

Figura 4.1 Exemplo de aplicação do algoritmo 2-opt.

Uma vez definido o algoritmo 2-opt, pode-se definir o algoritmo 3-opt, citado por Morawek (2003) como “uma continuação direta do algoritmo 2-opt”. Este método, usa princípios semelhantes à otimização 2-opt, com o diferencial de que são alterados três conexões, e não mais duas. Um exemplo desta operação é mostrado nas figuras 4.2(a) (caminho original) e 4.2(b) (caminho alterado).



(a) (b)

Figura 4.2 Exemplo de aplicação do algoritmo 3-opt.

Da mesma forma que o algoritmo 3-opt foi obtido através da evolução do algoritmo 2-opt, pode-se obter um algoritmo mais genérico denominado algoritmo λ-opt. A otimização λ é definida como sendo:

"Um caminho é dito λ-opt se for impossível obter um caminho menor substituindo quaisquer de suas ligações por outro conjunto de conexões" (Helsgaun, 1998).

Segundo Helsgaun (1998), "a generalização deste simples princípio forma a base de um dos mais eficientes algoritmos de aproximação para a resolução do TSP simétrico, o algoritmo de Lin-Kernighan."

É fácil notar que, quanto maior o número de cidades, maior o número de iterações necessárias para que a condição acima seja satisfeita. Lin e Kerninghan, conseguiram diminuir o custo computacional deste algoritmo ao incluir um λ-opt variável (Lin e Kerninghan, 1973). Com isso, a cada iteração o algoritmo altera o valor de λ, visando o menor caminho. A cada iteração, o algoritmo considera um conjunto maior de elementos a serem trocados. Este algoritmo continua sua execução até que uma condição de parada seja satisfeita (Helsgaun, 1998).

O autor enumera algumas das características principais deste algoritmo:

- sendo n o número de cidades, o algoritmo possui um tempo de execução na ordem de n^{22} (Helsgaun, 1998);

- é capaz de determinar a solução ótima para a maioria dos problemas com menos de 100 cidades.

O esforço de codificação necessário para a implementação do algoritmo de busca local de Lin-Kernighan, é comentado em Helsgaun (1998) que afirma: “os autores mostram que nenhuma outra implementação do algoritmo realizada até o momento havia mostrado uma eficiência tão boa quanto a obtida por Lin e Kernighan”.

3.2.4. Algoritmos genéticos

A teoria de algoritmos genéticos foi desenvolvida inicialmente por John Holland, se baseando no processo evolucionário encontrado na natureza (Tomassini, 1995). O algoritmo genético é baseado em um procedimento iterativo que age sobre uma população de indivíduos (soluções), cada uma representando uma solução possível no espaço de busca. A cada ciclo do procedimento, estes indivíduos são classificados segundo uma função de aptidão (*fitness*). Os indivíduos mais aptos ou seja, os que possuem maior aptidão, têm mais chances de passarem pelo operador de cruzamento (*crossover*) e probabilisticamente têm maior chance de sobreviver na próxima geração.

Em resumo, a heurística dos algoritmos genéticos pode ser detalhada pelas seguintes etapas:

Gerar aleatoriamente uma população inicial de indivíduos
Para cada indivíduo, obter o fitness de acordo com uma função custo
Enquanto um critério de parada não for satisfeito
{
    Selecionar os melhores indivíduos para reprodução
    Recombinar os indivíduos
    Aplicar o operador de mutação
    Avaliar o fitness de cada indivíduo
    Gerar uma nova população
}

Assim, se percebe que os algoritmos genéticos são estocásticos e baseados em população. O desempenho de um algoritmo genético depende dos operadores (Deb, 1999; Morawek, 2003):

- Reprodução/seleção;
- Cruzamento;
- Mutação.

A reprodução objetiva enfatizar as melhores soluções (candidatos) e penalizar as piores soluções de uma população. Para isso, escolhe-se os melhores resultados obtidos (normalmente aqueles classificados como sendo acima da média), realizam-se cópias dos resultados e substitui-se os piores resultados pelos novos. Deb (1999) cita vários métodos para que se realizem estas operações: "algumas abordagens usuais são a seleção por torneio, a seleção proporcional, a seleção por *ranking*, entre outros."

Uma rápida análise mostra que a reprodução não consegue gerar novas soluções para o sistema (Deb, 1999). Para isso, são utilizados os operadores de cruzamento (*crossover*) e mutação.

Deb (1999) analisa o uso do operador de cruzamento: "como o operador de reprodução, existem na literatura uma série de operadores de crossover, mas na grande maioria deles, duas *strings* (representando o genoma) são obtidas dos indivíduos vencedores e tem uma porção trocada entre si.". Morawek (2003), caracteriza o propósito deste operador como sendo a forma do algoritmo conservar o material genético de uma geração para outra e ao mesmo tempo combiná-lo de forma proveitosa visando a busca em novas áreas do espaço de busca.

O operador de mutação é o responsável pela geração da diversificação da população (Deb, 1999). Sem ele, o cruzamento e a reprodução logo convergiriam e obteriam o melhor elemento da população inicial, não do problema em si. Sua implementação consiste na mudança de alguma característica de um indivíduo, normalmente através de uma função probabilística.

Deb (1999) resume a ação destes três operadores: "o operador de reprodução/seleção seleciona os melhores resultados e o operador de *crossover* recombina duas partes dos resultados esperando assim obter um melhor resultado. O operador de mutação altera o resultado localmente esperando assim criar um resultado ainda melhor."

Morawek (2003) exemplifica o uso dos operadores de mutação e *crossover* para a resolução do problema do caixeiro viajante. De acordo com o autor, normalmente um gene representa uma lista de nós do caminho escolhido. O operador de mutação pode ser aplicado de várias formas, entre as quais o autor cita:

- troca de nós de um caminho (figura 4.3(a));
- rotacionamento de um *subtour* (trecho de caminho) (figura 4.3(b)).

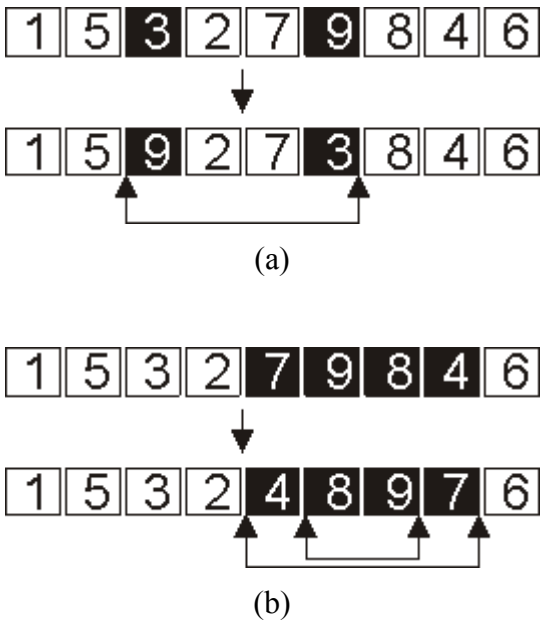


Figura 4.3 Aplicação do operador de mutação (Morawek, 2003).

A operação de *crossover* é exemplificada por Morawek (2003), onde um conjunto de nós é obtido do primeiro gene, e os demais nós são obtidos do segundo gene, mantendo sua ordem. Esta operação é exemplificada na figura 4.4.

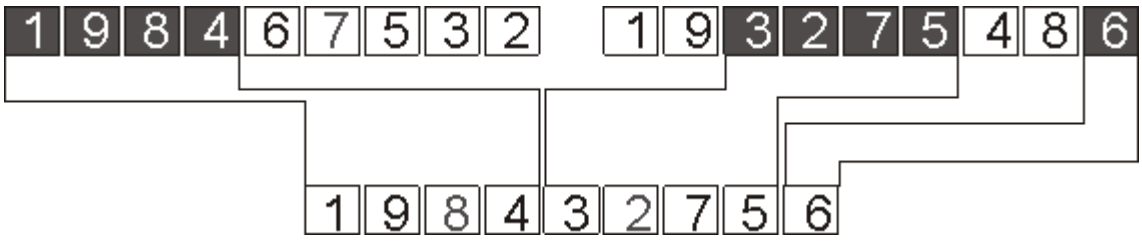


Figura 4.4 Aplicação do operador de *crossover* (Morawek, 2003)

Esta teoria é uma alternativa viável, na maioria das vezes, para vários problemas, como seleção de rotas dinâmicas (Kanoh e Nakamura, 2000), reconhecimento de padrões (Lavine *et al*., 2003), planejamento de sistemas produtivos, descobrimento de regras (Romão, 2002), planejamento de sistemas de potência (Coury, 2002), estudo de colônias de insetos (Lavine *et al.*, 2003), geração de rotas para robôs (Morandi e Nazemi, 2003), entre outros.

4.2.5. Programação evolutiva

Assim como algoritmos genéticos, a programação evolutiva (PE) utiliza conceitos de evolução para gerar soluções apropriadas através de um algoritmo iterativo (Coello, 2002). Embora seu pseudo-código seja semelhante ao dos algoritmos genéticos, existe uma diferença conceitual entre o alvo da análise da *fitness* dos indivíduos produzidos através de AG e PE. Tal diferença é explicitada em Coelho e Coelho (1999):

"A PE, bem como as estratégias evolutivas (...) diferem dos AGs, pois são técnicas que simulam a evolução enfatizando a ligação comportamental (relação fenotípica) entre as populações geradas (ancestrais e descendentes), mais que a ligação genética."

4.3.6. Otimização por colônia de formigas e outros modelos baseados em inteligência coletiva

Kelly (1995) define duas formas extremas de abordar um problema. A primeira, como uma seqüência de operações, da mesma forma que é realizada a produção em uma linha de montagem. A outra, como um grande conjunto de "operações paralelas", como as encontradas em neurônios ou nas operações realizadas por cada formiga de uma colônia. Ele define este último, onde "emergem do coletivo não uma série de ações individuais críticas, mas sim uma coleção de ações simultâneas aonde o padrão coletivo é de longe mais importante" como sendo o "modelo de enxame" (*swarm model*).

Para Kelly (1995), as duas abordagens dos problemas – através de um procedimento sequencial ou de um paralelo (mencionadas no parágrafo anterior) – são válidas. “(...) todos os sistemas da vida são misturas destes dois extremos.”

Os modelos de inteligência coletiva são baseados em dois ou mais agentes que comunicam para conseguir um comportamento emergente (Kazadi, 2000). Estes modelos, buscam se focar no comportamento de sistemas naturais constituídos por vários agentes, como colônia de bactérias, colônias de formigas, enxame de peixes e bando de pássaros (Martinoli e Easton, 2002). Existem outros exemplos de aplicações de sistemas cooperativos que imitam o funcionamento do sistema imunológico (Taheri e Calva, 2001). Ojala (1998), apresenta uma estratégia para controle de robôs inspirado em colônias de protozoários. Bourjot (2003) utiliza colônias de aranhas sociais para detecção de regiões em imagens. Conforme é exemplificado a seguir, no caso de colônia de formigas, as habilidades de tais sistemas parecem transcender as habilidades individuais de cada um dos componentes. É na interação entre os indivíduos da população que o comportamento que desejamos emerge.

Para este trabalho, é usada a definição de agente realizada por Franklin (1996): “Um agente autônomo é um sistema situado em e como parte de um meio que percebe este meio e atua nele, durante o tempo, visando seus próprios objetivos de forma a mudar sua percepção futura do meio”.

O conceito de comunicação adotado neste trabalho é o apresentado por Kazadi (2000) como “qualquer ação de um agente que influencie o comportamento subseqüente de um ou mais agentes do sistema seja esta influência direta ou indireta”.

O conceito de comportamento emergente é definido por Kazadi (2002) como: “um comportamento do enxame que seja resultado das interações do enxame com o meio ou entre si, mas que não seja resultado de um projeto direto.” O autor vai além, definindo que “o comportamento emergente requer comunicação entre os agentes, não necessariamente comunicação direta”. Brooks (1991), resume o conceito de comportamento emergente se um sistema multi-agente como “a inteligência emerge da interação dos componentes do sistema”. Esta afirmação também é sustentada por Zhong e Evans. (2002), que afirma “sem um líder ou controle centralizado, uma colônia de insetos resolve problemas que estão além da capacidade de um único inseto, como achar comida ou construir a colônia. A

inteligência dos grupos de insetos sociais ocorre por causa de suas interações alcançadas com a alteração e a detecção da concentração de feromônios químicos no ambiente."

Existem várias técnicas que se utilizam do conceito de inteligência coletiva. Kazadi (2002) cita várias pesquisas, como o mapeamento coletivo de terreno proposto por Cohen (1996), detecção de minas (Gage, 1995), construções através de conjuntos de robôs (Parker *et al.*, 2003) e a resolução de problemas do caixeiro viajante (Dorigo *et al.*, 1996).

O algoritmo da colônia de formigas foi apresentado inicialmente por Dorigo *et al.* (1996), propondo um paradigma computacional para a resolução de problemas combinatórios baseado no comportamento encontrado ao observar formigas se movendo entre a colônia e o ninho. Como indicado por Kelly (1995), este algoritmo não possui uma inteligência centralizada. Conforme é mostrado a seguir, a inteligência da colônia de formigas que resolve o problema não emerge de um indivíduo, mas sim da soma de todas as experiências de todos os indivíduos da colônia. Fox (2004) evidencia esta característica ao dizer "se você deseja entender a inteligência de enxames não olhe para uma abelha ou formiga específica. Você observa padrões de interação que emergem em um nível macro, inclusive aqueles gerados no nível micro aonde tudo parece caótico".

Algoritmos como este, inspirados em comportamentos de insetos sociais, vêm permitindo a geração de técnicas de otimização que geram resultados "surpreendentes" (Bonabeau *et. al.*, 2000).

Dorigo *et al.* (1996) e Lerman *et al.* (2001) enumeram algumas características adicionais do algoritmo de colônia de formigas:

- escalável, pois a arquitetura de controle de um robô é a mesma de algumas unidades ou milhares de unidades (Lerman *et al.*, 2001);
- versátil, pois pode ser usado por várias versões do mesmo problema; por exemplo, uma variação do problema do caixeiro viajante conhecida como sendo o problema assimétrico do caixeiro viajante. (Dorigo *et al.*, 1996; Lerman *et al.*, 2001);
- robusto, podendo ser aplicado com mudanças mínimas à outros problemas de otimização combinatória como o de atribuição quadrática, o escalonador de turnos de loja (Dorigo *et al.*, 1996; Lerman *et al.*, 2001) e planejamento de trajetórias de robôs móveis (Morawek, 2003). Como se trata de um sistema multi-agente, é pertinente citar Koening (1999), que afirma que sistemas de

busca baseados em agentes são "métodos de planejamento de propósito geral (independentes de domínio)", podendo "lidar com incerteza, incluindo ruídos de sensores e atuadores.". A robustez também é relacionada à técnicas de enxames por Pettinaro *et al.* (2002).

- dependente de população, permitindo a exploração da realimentação positiva como um mecanismo de busca (Dorigo *et al.*, 1996).

Pettinaro *et al.* (2002), afirma que técnicas de enxames são baseadas na auto-organização através da cooperação entre seus componentes. Desta forma, enxames de robôs possuem uma grande plasticidade, sendo seu uso indicado em quaisquer aplicações que necessitem de pouca intervenção humana e mecanismos de adaptação.

Tal algoritmo tem sido usado em pesquisas visando a otimização na gerência de redes de comunicação, conforme trabalho de Bieszczad *et al.* (1998). Bonabeau *et al.* (1998), também se utiliza deste algoritmo para o roteamento de pacotes de comunicação, explorando as características dinâmicas e descentralizadas do ACS. Parpinelli *et al.* (2002) apresentam uma aplicação deste algoritmo no campo de mineração de dados (*data mining*). Boryczka e Wieczorek (2003), mostraram o uso deste algoritmo para a regressão de funções em problemas de aproximação.

A fundamentação teórica do algoritmo descrito por Dorigo *et al.* (1996), se encontra na capacidade das formigas de gerar uma "trilha" entre o ninho e uma fonte de comida. Boryczka e Wieczorek (2003), evidenciam a inspiração biológica deste algoritmo ao citar três exemplos de características do ACS: (a) a escolha da rota de uma formiga artificial depende da quantidade de feromônio – uma substância química depositada pela formiga, (b) formigas artificiais cooperam visando obter melhores resultados, e (c) formigas artificiais se movem de uma forma aleatória.

A iteração entre os agentes do algoritmo da colônia de formigas é explicado por Bevilacqua e Castano (1999): "comportamentos globais complicados podem resultar da interação de sistemas dinâmicos individuais simples. Esses comportamentos podem ser observados em vários sistemas biológicos, como, por exemplo, em colônias de insetos sociais".

A visão tradicional sobre o comportamento desses insetos enxerga-os como pequenos seres autômatos que obedecem a um programa genético estritamente

estabelecido. Assim, supõe-se que os insetos possuem uma quantidade de "inteligência" individual suficiente para cooperarem de modo organizado. Essa visão se origina da idéia de que atividades complicadas estão necessariamente associadas a indivíduos complicados. Atualmente, uma nova interpretação vem ganhando força, segundo a qual os elementos aleatórios do meio ambiente e a maleabilidade do comportamento individual desempenham papéis fundamentais na organização do funcionamento da sociedade (Deneubourg *et al.*, 1983).

Talbi *et al.* (2001) confirmam a simplicidade do agente ao resumir como um conjunto de formigas consegue determinar o menor caminho entre cada ponto:

"Enquanto percorrem seu caminho, as formigas deixam uma trilha de feromônio no solo. O papel desta trilha é guiar as outras formigas até o alvo. Para uma formiga, o caminho é escolhido de acordo com a quantidade de feromônio. Posteriormente, a quantidade desta substância química vai diminuindo com a ação do tempo e a quantidade de feromônio restante depende da quantidade de alimento encontrada e do número de formigas que se utilizaram desta trilha."

Esta forma de comunicação indireta (através do meio), é uma das características marcantes deste algorítmo: ao contrário de outros sistemas multi-agentes como por exemplo, o proposto por Ozaki *et al.* (1999) ou o trabalho de Byrne *et al.* (2002), um agente nunca se comunica com outro. Ele se comunica sim, mas através do meio.

A dinâmica deste algoritmo é ilustrada na figura 4.5. Com o objetivo de ir do ponto A ao ponto B, inicialmente as formigas vão tanto para o caminho da esquerda quanto para o caminho da direita (figura 4.5(a)). Posteriormente, com a evaporação do feromônio, a trilha da esquerda se torna mais reforçada, pois um número maior de formigas passa por ele em um menor espaço de tempo (figura 4.5(b)). Após algum tempo, o caminho da esquerda conterá uma fração dominante de feromônios, o que levará as formigas a escolhê-lo com maior freqüência.

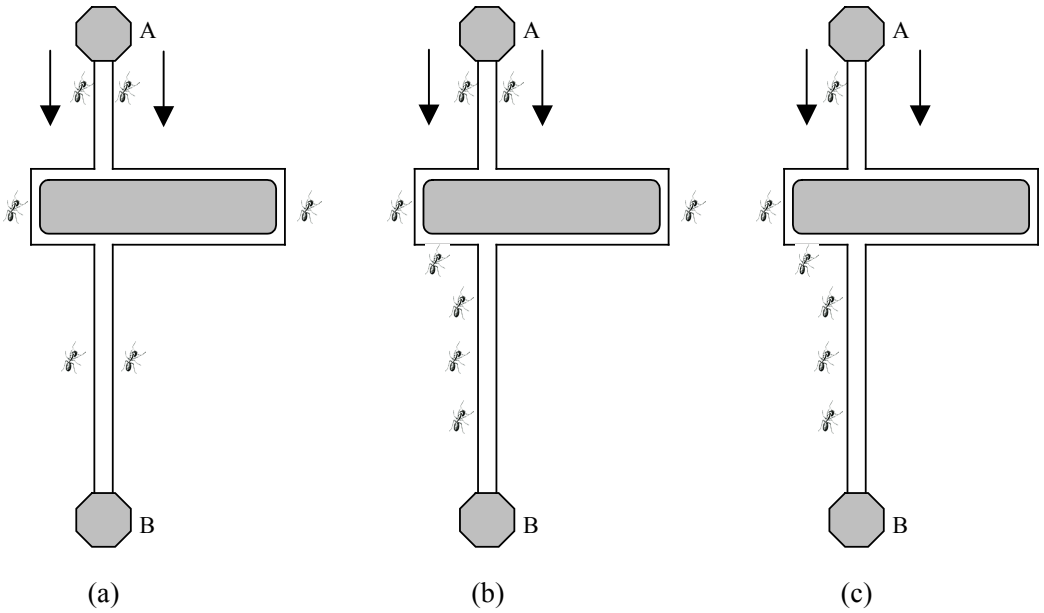


Figura 4.5 Descoberta do menor caminho através de informações de feromônio (Dorigo *et al*., 1996).

Dorigo e Gambardela (1997) aplicaram o algoritmo da otimização através da colônia de formigas no problema do caixeiro viajante, e compararam os resultados obtidos com outras técnicas como *algoritmos genéticos, programação evolucionária, simulated annaeling e annealing genetic algorithms*. Os resultados obtidos por Dorigo e Gambardela (1997) são reproduzidos na Tabela 4.1. Nesta tabela, foram aplicados os algoritmos mencionados em quatro problemas do caixeiro viajante, sendo registrados o tamanho do melhor percurso encontrado.

Os resultados apresentados por Dorigo mostram que o algoritmo da colônia de formigas como sendo recomendável para a resolução do TSP de grande porte.

Nome do Problema	ACS	AG	EP	SA	ANG	Melhor resultado
Oliver30 (30 cidades)	420	421	420	424	420	420
Eil50 (50 cidades)	425	428	426	443	436	425
Eil75 (75 cidades)	535	545	542	580	561	535
KroA100 (100 cidades)	21 282	21 761	N/A	N/A	N/A	21 282

Tabela 4.1 Resultados da aplicação de vários algoritmos distintos para problemas do TSP. (fonte: Dorigo e Gambardela, 1997).

Capítulo 5. Escolha do método

Nesta fase, a identificação de um método para direcionar a pesquisa da solução ao problema proposto no Capítulo 2. Os critérios utilizados nessa escolha foram decididos como sendo:

- a heurística a ser desenvolvida deve ser baseada em um algoritmo que produza um resultado eficiente;
- o algoritmo-base deve, de preferência, ser flexível o bastante para permitir mudanças futuras como o aumento de funcionalidades de ROVs.

Assim, com a análise qualitativa realizada no item 3.2, e levando-se em conta a análise quantitativa de Dorigo e Gambardela (1997), foi escolhido o uso do método da colônia de formigas.

Em uma primeira análise, mostrada em Tavares e Coelho (2003, 2004a), este método pareceu promissor. A seguir, é discutida a aplicação original do ACS em problemas do tipo caixeiro viajante, seguidas pelas modificações de Tavares e Coelho (2003), e, por fim, as modificações visando um aumento de desempenho do método.

5.1. Algoritmo ACS original

Para aplicar este comportamento ao problema do caixeiro viajante, Dorigo *et al.* (1996) e Dorigo e Gambardela (1997), utilizam-se a seguinte abordagem:

Seja $b_i(t)$ (i=1, ..., n) o número de formigas em um nó de um grafo, ou, no caso desta dissertação, intersecções de caminhos i no tempo t e $m = \sum_{i=1}^{n} b_i(t)$ é o número total de formigas no tempo t.

Seja cada formiga um agente simples com as seguintes características:

- capacidade de escolher a próxima cidade j levando em consideração a distância e a quantidade de feromônio presente no caminho entre as cidades (também chamadas de nós do grafo, ou, especificamente no caso da robótica de inspeção, objetivos a serem alcançados) i e j.

- uma lista tabu, que proíbe uma formiga de visitar a mesma cidade duas vezes;
- uma vez que a formiga cumpriu seus objetivos – visitar todas as cidades – deposita uma substância (feromônio) em cada caminho percorrido.

Boryczka e Wieczorek (2003), também citam que, no ambiente onde as formigas artificiais atuam, o tempo é discreto.

Dorigo *et al.* (1996) também modelaram o comportamento do agente, mensurando a probabilidade de escolha da ida de um ponto *i* para outro ponto *j* como sendo representada pela equação:

$$p_{i,j}^{k}(t) = \begin{cases} \dfrac{[\tau_{ij}(t)]^{\alpha} \cdot [\eta_{ij}]^{\beta}}{\sum_{k \in permitido_k} [\tau_{ik}(t)]^{\alpha} \cdot [\eta_{ik}]^{\beta}}, \text{se } j \in \text{permitido}_{\text{k}} \\ 0, \text{caso contrário} \end{cases} \quad (5.1)$$

onde $\tau_{ij}(t)$ representa a quantidade de feromônio existente no caminho entre as cidades *i* e *j* no instante *t*, η_{ij} representa a visibilidade da cidade *j* em *i* (definida como sendo o inverso da distância entre *i* e *j*), α e β são constantes, tais que $\alpha, \beta \in \mathrm{R}$.

Além disso, Dorigo *et al.* (1996) definiu que o feromônio é atualizado seguindo a equação (5.2). O termo $\rho \cdot \tau_{ij}^{k}(t)$ representa a parcela referente à evaporação, sendo ρ uma constante de evaporação tal que $0 < \rho < 1$. O termo $\Delta\tau_{ij}^{k}$ se refere à quantidade de feromômio depositada pela formiga entre os instantes *t* e *t* + *n*.

$$\tau_{ij}^{k}(t+n) = \rho \cdot \tau_{ij}^{k}(t) + \Delta\tau_{ij}^{k} \quad (5.2)$$

A quantidade de feromônio depositada pela formiga *k* em um caminho é definida na equação (5.3), aonde *Q* representa uma constante e *L* é o comprimento total percorrido pela formiga,

$$\Delta\tau_{ij}^{k} = \begin{cases} \dfrac{Q}{L_k} \text{ se a formiga } k \text{ utiliza desta trilha em seu caminho} \\ 0, \text{ caso contrário} \end{cases} \tag{5.3}$$

É razoável, então, que a quantidade de feromônio depositada por todas as *m* formigas em um caminho de *i* a *j* possa ser definida como:

$$\Delta\tau_{ij} = \sum_{k=1}^{m} \Delta\tau_{ij}^{k} \tag{5.4}$$

Baseado no proposto por Dorigo *et al.* (1996), o algoritmo original do ACS está mostrado na Figura 5.1.

1. Configuração do Algoritmo:
   t=0, NC=0 (t representa o tempo, NC representa o número de ciclos)
   Para cada caminho (i,j), selecione um valor inicial $\tau_{ij}(t)=c$ para a intensidade do
   feromônio e $\Delta\tau_{ij}= 0$
   Coloque m formigas nos n nós
2. s=1 (s representa o índice da lista tabu)
   Faça de k=1 a m
   Selecione o ponto inicial da *k-ésima* formiga em **tabu**(s)
3. Repita até que a lista tabu esteja cheia (n-1 vezes)
   s=s+1
   Faça de k=1 a m
   Escolha o ponto j para mover a formiga, com a probabilidade dada pela equação (3.2).
   Mova a *k-ésima* formiga para o ponto j
   Insira o ponto j na lista Tabu
4. Faça de k=1 a m
   Mova a *k-ésima* formiga de *tabu(n)* a *tabu(*1*)*
   Determine o comprimento L_k do caminho percorrido
   Atualize o feromônio do último caminho encontrado, de acordo com a equação (3.4).

Figura 5.1 O algoritmo da colônia de formigas de acordo com Dorigo *et al.* (1996).

5.2. Análise do algoritmo da colônia de formigas aplicado ao problema de planejamento de rotas de ROVs com cordão umbilical

Para a utilização do ACS para a solução do problema definido nesta pesquisa, são estabelecidas algumas novas heurísticas (Tavares e Coelho, 2003, 2004a).

- a primeira diferença aparece no objetivo do sistema: embora no problema original se tenha como alvo passar por cidades – que agora são tratadas como intersecções de caminhos – neste caso, o alvo são os caminhos entre as intersecções;
- no TSP original, o objetivo é a passagem por todas as cidades. Neste problema, pode-se ter rotas que não necessitam de inspeção. Assim, nem todos os caminhos existentes serão necessariamente os objetivos do problema;

- o cordão umbilical é mais uma limitação de movimento, conforme apresentado na seção 2.2.

A seguir, são mostradas as alterações realizadas inicialmente no algoritmo ACS apresentado por Dorigo *et al.* (1996) visando a solução deste problema.

Representação do cabo umbilical

Para a representação do cabo umbilical, utilizou-se o conceito de uma pilha FILO (*First In Last Out* – Primeiro elemento a entrar é o último a sair) (Taub, 1984), aplicando as seguintes regras:

- *C(q)* representa o vetor correspondente aos elementos da pilha FILO;
- $q_{\max}$ é o número máximo de elementos que a fila pode possuir;
- q_{atual} representa o número atual de elementos da fila.

O conceito da lista FILO é representado na figura 5.2, onde são exemplificados uma fila vazia com q_{max}=3 (figura 5.7(a)), a inclusão de 1, 2 e 3 itens (figuras 3.8(b), (c) e (d), respectivamente) e a remoção de um item (figura 5.2(e)).

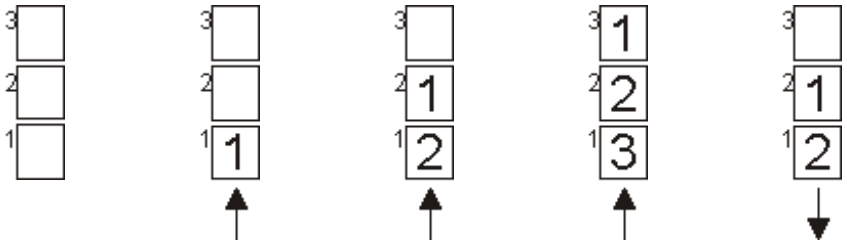


Figura 5.2 Exemplo de operação de uma pilha FILO.

A cada movimento da formiga, insere-se o novo destino no vetor *C(q)*. Posteriormente, devem-se tomar medidas para evitar que $q_{atual} > q_{\max}$.

Fim da lista tabu

Em casos de inspeção, é possível a necessidade de se passar pelo mesmo ponto várias vezes. Assim, a lista tabu foi, em uma primeira instância, removida.

Mudança na regra de movimento

Com a definição do cabo umbilical e o fim da lista tabu, deve-se redefinir a regra que rege a determinação dos possíveis destinos para a formiga:

Se o único caminho existente for o caminho de volta, a formiga passa por este caminho. Caso contrário, verifica-se o estado do cabo:

Se $q_{atual} = q_{max}$, o único caminho possível será o caminho anterior;

Se $q_{atual} < q_{max}$, os caminhos possíveis são todos menos o caminho anterior.

É importante notar que a condição $q_{atual} > q_{\max}$ nunca ocorre. Isso se deve ao fato de que, quando tomamos o caminho de volta forçadamente devido à condição $q_{atual} = q_{max}$, retiramos um elemento da lista FILO, consequentemente decrementando q_{atual}.

Critério de parada

Sendo C o conjunto com todos os caminhos percorridos, i representa o número de elementos deste conjunto, t é o tempo em que a formiga está em operação, $t_{\max}$ é o tempo máximo de vida de uma formiga e O é o conjunto de objetivos da formiga:

Se $O \subset C$: formiga completou os objetivos, e tem o feromônio de seu caminho atualizado;

Se $t \geq t_{max}$: a formiga "morre" e seu feromônio não é adicionado no meio.

Resultados obtidos

O algoritmo apresentado anteriormente foi implementado em JAVA. Foi proposto um grafo de ensaio com 11 nós, conforme mostrado na Figura 5.3. Nesta figura, os círculos são nós, os traços caminhos e o campo "Início" representa o ponto de saída da formiga.

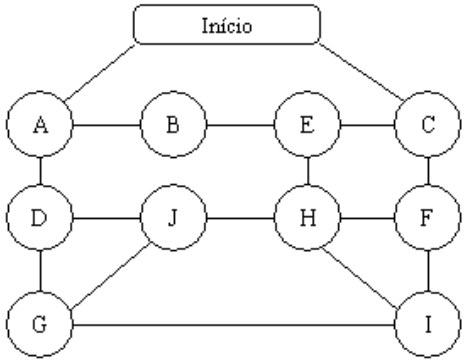


Figura 5.3 Grafo de ensaio com 11 nós (Tavares e Coelho, 2003).

Selecionou-se como objetivo a passagem pelos nós D e H, e com uma limitação devido à corda de 10 movimentos. Foram enviadas 22 formigas sequencialmente a partir do ponto marcado "início" (empiricamente, se determinou que, para este caso, a relação de duas formigas para cada nó do problema é suficiente para a convergência do algoritmo). Após alguns testes com o algoritmo implementado, os demais parâmetros foram determinados empiricamente como sendo:

- $\alpha = 1$;
- $\beta = 0$ (com isso, o termo referente à visibilidade dos nós, considerado constante no problema é ignorado);
- $\rho = 0{,}9$.

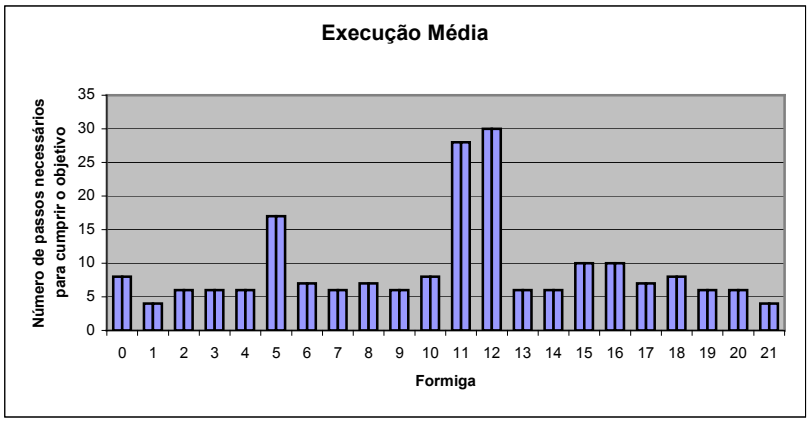
Nas Figuras 5.4(a) e 5.4(b) têm-se, respectivamente, os caminhos percorridos e uma análise gráfica do número de passos necessários para que cada formiga atinja os objetivos. Na Figura 5.4(a), tem-se os dados apresentados no formato $f[x_0,x_1,x_2,x_3,...,x_n]$, sendo que f é

o número da formiga (coincidente com a ordem que o agente foi enviado), e $x_{0...n}$ os nós visitados após o nó inicial.

A ação do cordão umbilical é destacado com os nós em negrito. Quando isso corre, existe o retorno imposto pelas condições de contorno já apresentadas. Conforme ilustrado na figura 5.4(a), a ação do cordão umbilical torna vários caminhos proibitivos devido ao retorno que causa na trajetória da formiga. Quando, mesmo tendo o cordão umbilical estendido, a formiga consegue atingir seu objetivo, o traço de feromônio é estendido para mais nós, causando um distúrbio no meio, percebido no gráfico da figura 5.4(b) quando analisamos os caminhos das formigas 11 e 13. Mesmo assim, conseguiu-se determinar o caminho ótimo para o problema.

0. [C, Início, C, F, H, I, G, D]
1. [A, D, J, H]
2. [A, B, A, D, I, H]
3.]C, E, H, I, G, D]
4. [C, F, H, I, G, D]
5. [C, F, C, E, H, F, C, E, H, I, G, I, H, E, C, Início, A, D]
6. [A, D, A, Início, C, F, H]
7. [A, Início, A, D, I, H]
8. [A, Início, C, F, H, I, G, D]
9. [A, B, A, D, I, H]
10. [C, Início, C, E, H, I, G, D]
11. [A, B, A, Início, C, E, H, F, C, E, H, E, C, Início, A, Início, C, F, H, E, C, Início, A, B, E, H, I, G, D]
12. [C, F, H, E, C, Início, A, B, E, B, A, Início, C, F, H, I, G, I, H, E, C, E, H, F, C, E, H, I, G, D]
13. [A, D, A, Início, C, F, H]
14. [A, D, A, B, E, H]
15. [A, B, A, Início, C, F, H, I, G, D]
16. [C, Início, C, E, H, F, C, Início, A, D]
17. [A, D, A, Início, C, F, H]
18. [C, F, C, E, H, I, G, D]
19. [A, D, A, B, E, H]
20. [A, D, A, B, C, F]
21. [A, D, J, H]

(a)



(b)

Figura 5.4 Resultados obtidos (Tavares e Coelho, 2003).

Este resultado típico mostrou que o algoritmo possui a capacidade de obter o menor caminho rapidamente (neste caso, foi obtido já na segunda execução). Porém, outras execuções encontraram caminhos diferentes (e de comprimento maior), que geravam trilhas de feromônios em pontos fora da melhor rota. Tal comportamento, embora seja entendido quando se analisa a equação (5.1) que define a movimentação da formiga através de um procedimento aleatório, causou distúrbios pontuais no resultado da atuação de cada agente no meio. Nota-se, na execução mostrada na Figura 5.4(b), que as formigas 5, 11 e 12 escolheram caminhos muito maiores (mais de três vezes maior) que o caminho ótimo do algoritmo. Este comportamento foi notado em outras execuções em diferentes formigas, o que causou uma dificuldade na análise de várias instâncias do problema. Assim, escolheu-se utilizar uma execução em que se observou o comportamento médio do algoritmo. Desta forma, o algoritmo mostrou a necessidade de que exista uma otimização nas buscas pelos mínimos globais e locais do problema.

Os resultados sugerem a necessidade de que seja controlada a influência de agentes que consigam obter o caminho mínimo através de caminhos menos otimizados que os já encontrados.

Visando realizar uma análise dos algoritmos que permitam suprir estes requisitos, foram determinadas mudanças no conjunto de heurísticas adotadas tendo em vista melhorar sua convergência: um reforço negativo baseado na memória individual de cada formiga e a restrição do espaço de busca pelos próprios agentes através de algoritmo cultural. Para isso, foi analisado apenas o problema sem o cordão umbilical e sem a lista tabu. A heurística resultante e os resultados são mostrados a seguir.

5.3. Adaptação do ACS visando melhorar a convergência

O algoritmo ACS original, não permite o planejamento de múltiplos objetivos quando estes compartilham um mesmo caminho. As passagens duplicadas em um mesmo corredor são proibidas pela lista tabu do algoritmo (Dorigo *et al.*, 1996). Tal efeito, pode ser desastroso quando é necessário, por exemplo, voltar por um mesmo caminho (caminho de inspeção interrompido).

Assim, objetivando solucionar este problema, suprimiu-se a lista tabu. No seu lugar, foram colocados três mecanismos de controle: um nível mínimo inicial de reconhecimento de feromônio (*feromBias*), um parâmetro relacionado à memória da formiga (*QBad*), e um nível mínimo de reconhecimento final de feromônio, após a ação do reforço negativo (*feromBiasFinal*). A ação do reconhecimento do feromônio pelo agente, é substituída pela heurística apresentada na Figura 5.5. Esta heurística, se utiliza de duas equaçãoes: a equação (5.5), aplica uma penalização no valor de feromônio de um nó de *QBad* unidades para cada vez que a formiga tenha passado nesse nó. Para evitar valores negativos, a equação também garante que o patamar mínimo de feromônio, *feromBias*, será respeitado. A equação (5.6), garante que um segundo patamar mínimo é respeitado, evitando assim que o valor de feromônio de um ponto seja menor que o mínimo perceptível pela formiga.

1. Determine quais os caminhos adjacentes válidos e seu valor de feromônio
2. Normalize o valor do feromônio utilizando a equação (3.5), sendo x o feromônio da posição analisada
3. Sendo *count(n)* uma função que retorna o número de vezes que a formiga passou pela posição n e *g(n)* a função que mostra a quantidade de feromônio após a atuação do parâmetro relacionado à memória da formiga, normalize o feromônio utilizando a equação (5.5).
4. Normalize o valor do feromônio utilizando a equação (5.6)

Figura 5.5 Heurística para reconhecimento do feromônio.

$$f(x) = \begin{cases} x - QBad \cdot count(x), \text{se } x > feromBias \\ feromBias - QBad \cdot count(x), \text{se x} \leq feromBias \end{cases} \quad (5.5)$$

$$g(x) = \begin{cases} x, \text{se } x > BadFeromBias \\ BadFeromBias, \text{se } x \leq BadFeromBias \end{cases} \quad (5.6)$$

A seguir, algumas análises usando este algoritmo são apresentadas. Inicialmente, o ACS com reconhecimento de feromônio foi testado com o mapa mostrado na Figura 5.6. Este mapa, foi escolhido por ser o exemplo mais simples no qual poderia ser testado a convergência entre dois caminhos. Esta experiência, mostrada na Figura 5.6, possui apenas uma escolha entre duas rotas, não havendo interferência de outras decisões dos agentes.

Adicionalmente, as duas decisões influenciam de forma diferente o comprimento do caminho por onde a formiga passa, tornando assim simples sua análise. Para que se pudesse considerar apenas a variação do feromônio, considerou que cada quadrado da figura era um nó do grafo. Ao aplicar a equação (5.1), novamente se optou por β=0, eliminando assim o efeito deste parâmetro no algoritmo.

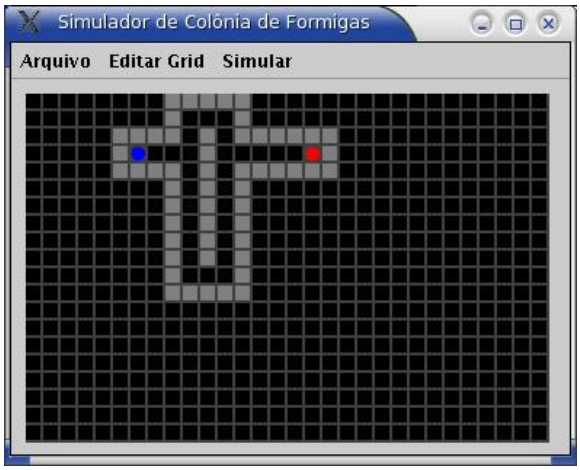


Figura 5.6 Mapa analisado após adicionar-se o reforço negativo.

5.3.1. Resultados obtidos através da aplicação do reforço negativo

Como o objetivo desta análise é verificar se existe uma melhor convergência com este algoritmo, gerou-se 100 execuções, sendo que cada execução com 10 agentes, mantendo-se os valores de α, β e ρ da execução anterior. Para cada execução, obteve-se o desvio padrão do número de passos. Como o desvio padrão é uma medida da dispersão dos dados em torno do valor médio, pode-se seguramente analisar o desvio padrão como sendo um índice ligado à convergência do algoritmo. Se todos os agentes forem em um só caminho, o desvio padrão é nulo. Se cada agente for por um caminho diferente, o desvio padrão alcança seu valor máximo.

Foram alterados isoladamente os parâmetros *BadFeromBias* e *Qbad*. Os resultados são mostrados nas Tabelas 5.1 e 5.2, respectivamente.

BadFeromBias	Média do desvio padrão do número de passos por execução	Desvio padrão do desvio Padrão do número de passos por execução
0,01	5,96	5,49
0,10	6,15	5,49
0,50	8,3	8,21
0,75	9,07	8,95
1,00	9,34	9,31

Tabela 5.1 Análise do movimento dos agentes com a alteração do parâmetro *BadFeromBias*.

Qbad	média do desvio padrão do número de passos por execução	desvio padrão do desvio padrão do número de passos por execução
0,01	9,2	9,33
0,10	9,17	9,46
0,50	7,11	7,33
0,75	7,45	6,64
1,00	6,64	5,53

Tabela 5.2 Análise do movimento dos agentes com a alteração do parâmetro *QBad*.

Nota-se que existe uma diminuição do desvio padrão com o aumento do valor da variável *Qbad*. Com isso, nesta análise, mostrou-se uma influência desta memória nas ações dos agentes e na eficiência da otimização. Ou seja, para esta situação estudada, quanto maior a influência da memória do agente, menor a chance de repetição de um caminho.

O aumento do valor de *BadFeromBias* causou um efeito contrário. Assim, notou-se que a convergência do algoritmo, neste caso, foi melhor quando a normalização imposta pela equação (3.5) foi menos influente. Este comportamento pode ser explicado pelo fato de que quando se toma um limite mínimo de feromônio alto, este encobre pequenas mas relevantes alterações de valores dos feromônios, não permitindo que as formigas recebam o reforço positivo previsto.

Buscando-se analisar padrões de comportamento, foi repetida a análise para o gráfico mostrado na Figura 5.7. Este gráfico possui um número maior de caminhos possíveis e redundantes. Enquanto o mapa mostrado na Figura 3.11 possui dois caminhos

diretos possíveis, o mapa mostrado na Figura 3.15 possui um número maior de rotas possíveis. Os resultados obtidos são mostrados nas Tabelas 5.2 e 5.3.

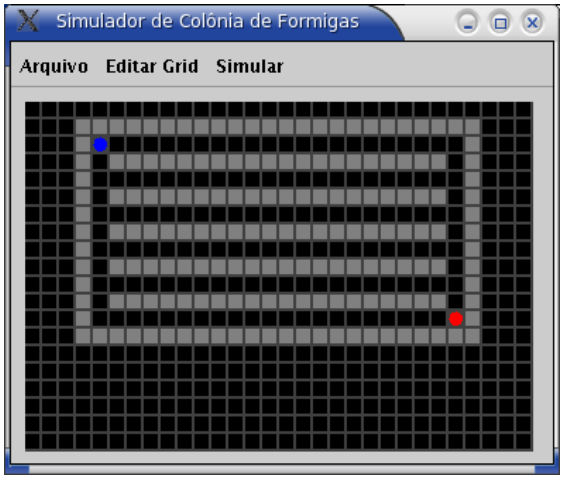


Figura 5.7 Mapa analisado com o algoritmo da seção 5.3.

BadFeromBias	média do desvio padrão do número de passos por execução	desvio padrão do desvio padrão do número de passos por execução
0,01	30,18	23,33
0,10	31,14	27,57
0,50	31,05	29,98
0,75	30,17	30,44
1,00	42,06	38,23

Tabela 5.3 Análise do movimento dos agentes com a alteração do parâmetro *BadFeromBias.*

Qbad	média do desvio padrão do número de passos por execução	desvio padrão do desvio padrão do número de passos por execução
0,01	37,08	28,64
0,10	38,47	23,60
0,50	33,33	24,93
0,75	32,44	33,76
1,00	32,93	34,12

Tabela 5.4 Análise do movimento dos agentes com a alteração do parâmetro *Qbad.*

Através dos resultados apresentados nas Tabelas 5.3 e 5.4, notou-se novamente que a variação entre os movimentos das formigas e os novos parâmetros, *BadFeromBias* e *Qbad* são relacionados, existindo um ponto de equilíbrio entre eles em que se consegue

uma menor divergência do algoritmo. Pelo que podemos notar, a variação de *Qbad* permitiu melhores resultados quando seu valor foi de 0,5, diferente do encontrado no exemplo da figura 5.5, onde o melhor resultado foi de 0,01. Com isso, conclui-se que a influência entre a variação de *Qbad* é estritamente relacionada com o grafo que o agente percorre, da mesma forma que os parâmetros α, β e ρ do algoritmo original.

5.4. Alteração do critério de parada para reforço do caminho "ótimo" encontrado

Sutton (1992) denomina o tipo de reforço apresentado no ítem anterior como de arquitetura de políticas somente (*policy-only architectures*). Esta arquitetura, ainda segundo Sutton (1992), requer que sejam claras as condições em que um alto reforço está presente (*high rewards*, normalmente de valor positivo) e um baixo reforço (*low rewards*, normalmente de valor negativo). No caso, analisando apenas o reforço referente à memória do agente, existe um reforço negativo na probabilidade de escolha de um nó proporcional ao número de visitas que o agente realizou neste nó. Analisando apenas o agente, sem contar sua comunicação através de feromônios, não há reforço positivo. Este reforço positivo ocorre apenas na análise dos feromônios mostrada nas seções 3.3.1 à 3.3.3.

Para a melhor atuação das heurísticas de reforço apresentadas, nota-se que é necessário substituir o critério de parada mostrado na seção 5.3 pelas regras a seguir, que reforçam ainda mais os caminhos ótimos encontrados durante a execução do algoritmo:

- se $O \subset C$: formiga completou os objetivos e tem o feromônio de seu caminho atualizado e
- se $t \geq t_{\max}$: formiga assume o caminho ótimo encontrado até o momento.

Com esta modificação, há o depósito de feromônio mesmo quando a formiga, de acordo com o critério anterior, reforça o melhor caminho encontrado até o momento.

5.5. Limitação do espaço de busca através de algoritmo cultural

O algoritmo apresentado até o momento, foi dotado de heurísticas que agiram sobre um indivíduo isoladamente. Porém, todas as alterações realizadas alteraram apenas o indivíduo.

Para aumentar a rapidez de convergência, foi notada a importância de aumentar as formas de cooperação entre os agentes, de modo que eles possam trocar informações sobre trechos já reconhecidos como não pertencentes ao caminho ótimo.

Com este fim, foi decidido usar utilizar variações do algoritmo cultural (Reynolds, 1997; Iacoban *et al.*, 2003). O algoritmo cultural foi usado por Jin e Reynolds (1999) para a solução de problemas de otimização não-linear. Reynolds e Zhu (2001) usaram algoritmos culturais em combinação com programação evolutiva. Ostrowski *et al.* (2002) evoluiram estratégias em modelos baseados em agentes para auxiliar o desenvolvimento de *software* usando algoritmos culturais.

O princípio do algoritmo cultural pode ser abordado através da definição de cultura. Segundo Reynolds e Zhu (2001), o termo cultura pode ser entendido como "um veículo de armazenamento de informações acessíveis globalmente por todos os membros da sociedade e que pode ser útil em direcionar suas atividades para solução de problemas". Na figura 5.8 se mostra o diagrama de funcionamento de um algoritmo cultural, onde temos uma população de agentes que se altera de acordo com uma função de performance, aceitando (e modificando) um espaço de crenças comum. Este espaço de crenças influencia toda a população, alterando assim sua evolução.

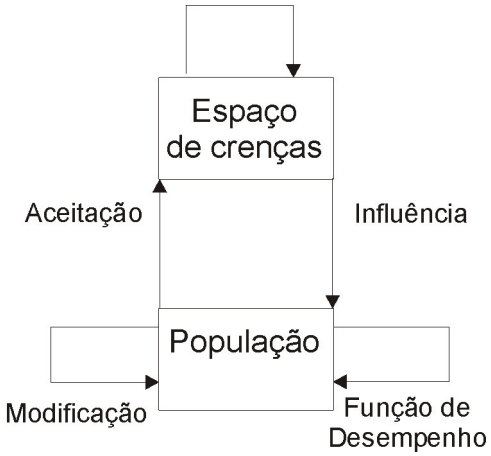
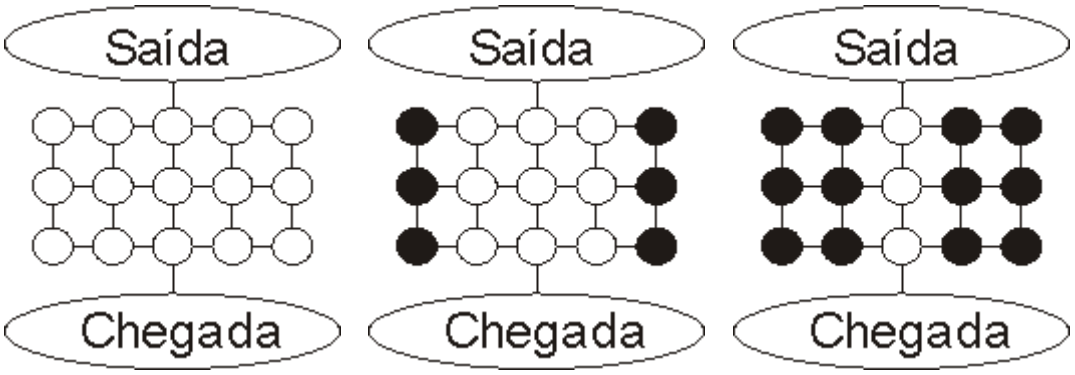


Figura 5.8 – Configuração de um sistema evolutivo com algorimo cultural (Reynolds e Zhu, 2001).

Os algoritmos culturais, então, surgem como um algoritmo de suporte. Através da troca de experiências entre os vários elementos de uma população de agentes, são alteradas as crenças de toda uma população. Estas crenças, muitas vezes referidas como espaço de crenças (*belief space*), são utilizadas para direcionar a evolução de toda a população e de sua ação no meio. Pode-se, tomando o exemplo do planejamento de rotas, utilizar o algoritmo cultural para reduzir o espaço de busca do agente, para que este possa concentrar-se em buscas locais. Na Figura 5.9, é apresentado um exemplo de agente que deve ir de um ponto a outro do espaço. Inicialmente, como mostrado na Figura 5.9 (a), o agente busca o menor caminho por todo o espaço. Em um segundo momento (Figura 5.9 (b)), o algoritmo cultural age, reduzindo o espaço de busca do agente (os nós em preto simbolizam os nós eliminados). Em um terceiro momento, onde o agente se utiliza apenas do menor caminho, o espaço de busca e o menor caminho são coincidentes (Figura 5.9 (c)).



(a) (b) (c)

Figura 5.9 Exemplo da alteração de um espaço de busca por um algoritmo cultural.

O pseudo-código do algoritmo cultural é mostrado na Figura 5.10.

```
Início
    t=0
    Inicialize a população POP(t)
    Inicialize o espaço de crenças BLF(t)
    Repita
        Avalie a população POP(t)
        Determine a população POP_Accepted(t) que contribui para o ajuste
de BLF(t)
        Ajuste BLF(t) de acordo com POP_Accepted(t)
        t=t+1
        Selecione POP(t) = POP(t-1)
    Até que o critério de parada seja alcançado
Fim
```

Figura 5.10 Pseudo-código de um algoritmo cultural.

Neste caso, tem-se os seguintes parâmetros:

POP(t) representa a população no instante *t;*

BLF(t) representa o espaço de crenças que se deseja alterar no instante *t;*

POP_Accepted(t) representa um subconjunto de *POP(t)*, contendo os indivíduos mais significativos para o ajuste do espaço amostral.

É interessante notar que as crenças são criadas e alteradas de acordo com a experiência de uma população, não de um indivíduo. Embora, em um momento inicial, todas as crenças possam ser as mesmas, espera-se que a interação desta população entre si e entre o meio em que está envolvido traga novas informações que diferenciem indivíduos o bastante para a formação da população *POP_Accepted*. E, através destas modificações, resultantes de experiências de cada indivíduo, é possível realizar ajustes em *BLF* de forma a guiar futuras evoluções da população.

Durante a pesquisa bibliográfica, não foi encontrada nenhuma abordagem que combinasse alguma abordagem baseada em colônia de formigas com algoritmo cultural. Para tal, foram tomadas as seguintes decisões de implementação visando a combinação do algoritmo apresentado nas seções 3.3.1 a 3.3.4 com o algorimo cultural:

(a) *BLF(t)* representa o conjunto de destinos (nós) que pertencem ao mapa;

(b) *POP(t)* representa a união de um conjunto de *c* colônias. Cada colônia é constituída por *f* formigas;

(c) *POP_Accepted(t)* representa um número majoritário de colônias que possuem um número majoritário de formigas que não utilizam um determinado nó em seus percursos;

(d) *c_min* representa o número mínimo de colônias que não devem utilizar um determinado nó para que a colônia pertença à *POP_Accepted(t);*

(e) *f_min* representa o número mínimo de formigas que não devem utilizar um determinado nó para que a colônia correspondente pertença à *POP_Accepted(t)*;

(f) *BLF(t+1)* representa o conjunto *BLF(t)* excluindo-se os elementos identificados por *POP_Accepted(t).*

Também deve-se notar as seguintes condições de contorno do problema:

(a) *c_min* deve ser sempre maior que a metade no número de colônias pertencentes à *POP(t).* Desta forma, assegura-se que, tendo *c* caminhos ligando dois nós, pelo menos um será mantido sempre.

(b) *c_min* deve ser sempre maior ou igual ao número de colônias pertencentes à *POP(t)*. Se *c_min* não respeitar esta condição de contorno, *POP_Accepted(t)* sempre é um conjunto vazio.

(c) Da mesma forma, *f_min* deve ser maior que a metade do número de formigas de uma colônia e menor ou igual à este valor.

O pseudo-código resultante é mostrado na figura 5.11.

```
Início
    t=0
    Inicie a população POP(t) com c colônias com f formigas cada.
    Inicie c_min e f_min
    Inicie o espaço de crenças BLF(t) com o mapa inicial
    Repita
        Execute uma vez cada formiga de cada colônia de acordo com o
        algoritmo já mostrado em itens anteriores
        Determine a população POP(t) como o conjunto de colônias
        executadas no instante de tempo t
        Para cada nó n pertencente à BLF(t)
                Faça F(n, c) igual ao número de formigas da colônia c
                pertencentes à POP(t) que não utilizam o nó n
                Faça POP_Accepted(t) igual ao número de colônias que
                possuem f_min ou mais formigas que não utilizam o nó n no
                tempo t
                Se C(n) for maior que c_min, remova o nó n de BLF(t).
        t=t+1
        Selecione POP(t) = POP(t-1)
    Até que o critério de parada seja alcançado
Fim
```

Figura 5.11 Pseudo-código do algoritmo cultural aplicado à colônia de formigas.

É importante notar as condições de contorno propostas neste trabalho representadas nas equações (5.7) e (5.8):

$$f \geq f\_\min > \frac{f}{2} \tag{5.7}$$

$$c \geq c\_\min > \frac{c}{2} \tag{5.8}$$

Tais condições, determinam que:

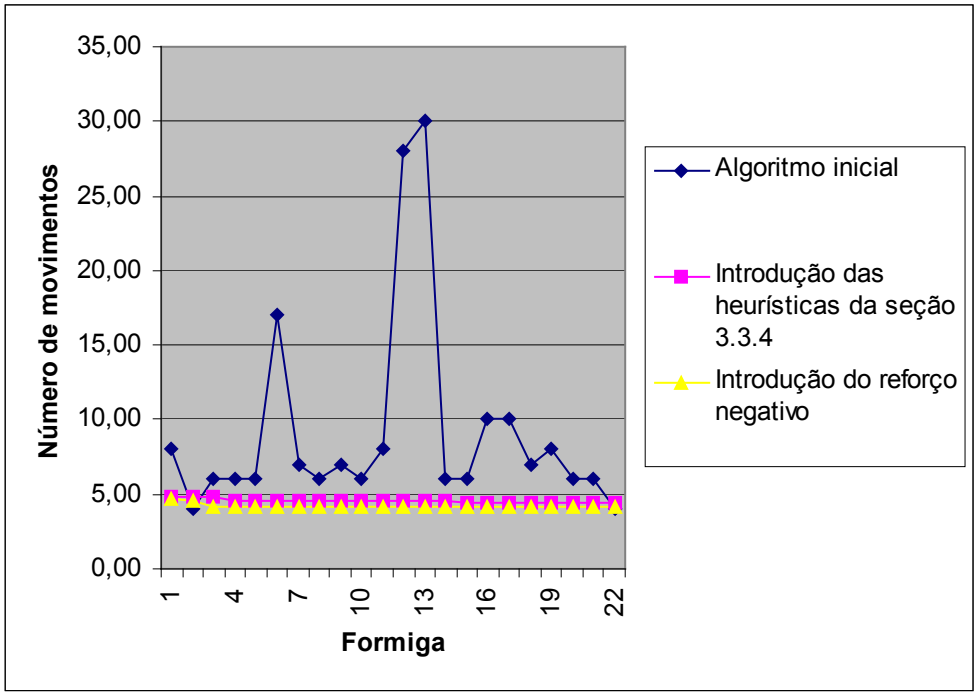
- *f_min* e *c_min* não devem ser maiores que o número total de formigas e de colônias respectivamente. Isso garante que, neste limite de operação, um nó é eliminado quando não for usado por nenhuma formiga (*f_min* = *f*) e por nenhuma colônia (*c_min* = *c*);
- *f_min* e *c_min* devem ser maiores que a metade do número de formigas e colônias respectivamente. Com isso, tem-se vários caminhos ligando dois nós onde passam as formigas, estes nunca são eliminados simultaneamente.

5.5.1. Resultados obtidos usando o algoritmo cultural

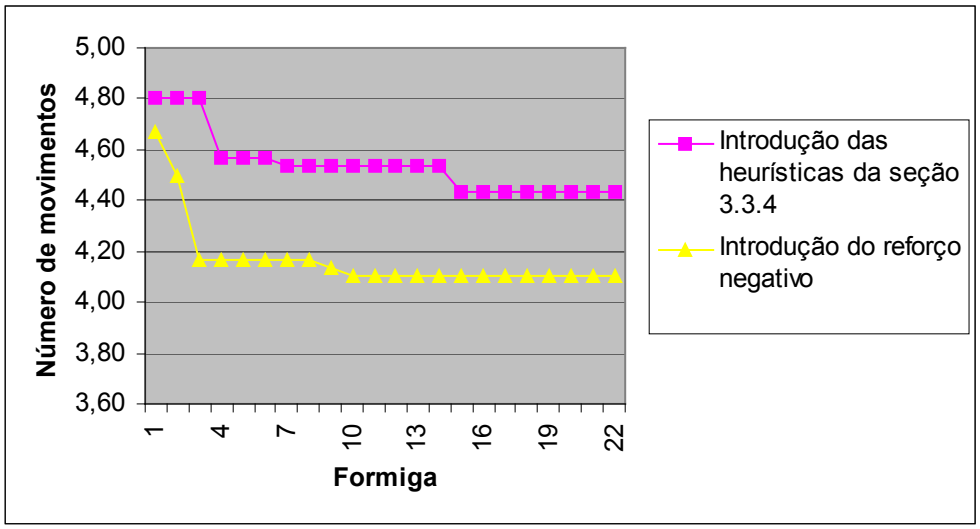
Tavares e Coelho(2005) realizaram uma análise do mesmo problema da seção 3.3.2, confrontando os dados encontrados com o respectivo algoritmo com dados obtidos através das variações das heurísticas até aqui descritas. O objetivo desta análise é determinar se, com um mesmo número de agentes, o algoritmo cultural traz benefícios na convergência do problema.

Para realizar esta tarefa, o número de formigas adotado foi mantido constante (27) e se buscou analisar o número de passos que cada formiga necessitou para completar seu objetivo. Ao introduzir a alteração mostrada na seção 5.4 e o reforço negativo, o algoritmo foi executado 30 vezes. Para o algoritmo mostrado na seção 5.2, tomou-se um resultado significativo e não uma média.

Foram encontrados os resultados ilustrados nas Figuras 5.12(a) e 5.12(b). Os dados correspondentes são mostrados na Tabela 5.5.



(a)



(b)

Figura 5.12 Heurísticas da seção 3.3.4 e do reforço negativo (seção 3.3.3) em comparação com o algoritmo mostrado na seção 3.3.2(a). Em (b), tem-se destacado apenas as heurísticas das seções 3.3.3 e 3.3.4.

Média de movimentos	Algoritmo inicial	Mudança do critério de parada	Introdução do reforço negativo
formiga 0	8,00	4,80	4,67
formiga 1	4,00	4,80	4,50
formiga 2	6,00	4,80	4,17
formiga 3	6,00	4,57	4,17
formiga 4	6,00	4,57	4,17
formiga 5	17,00	4,57	4,17
formiga 6	7,00	4,53	4,17
formiga 7	6,00	4,53	4,17
formiga 8	7,00	4,53	4,13
formiga 9	6,00	4,53	4,10
formiga 10	8,00	4,53	4,10
formiga 11	28,00	4,53	4,10
formiga 12	30,00	4,53	4,10
formiga 13	6,00	4,53	4,10
formiga 14	6,00	4,43	4,10
formiga 15	10,00	4,43	4,10
formiga 16	10,00	4,43	4,10
formiga 17	7,00	4,43	4,10
formiga 18	8,00	4,43	4,10
formiga 19	6,00	4,43	4,10
formiga 20	6,00	4,43	4,10
formiga 21	4,00	4,43	4,10

Tabela 5.5 - Séries de dados referentes à figura 5.12.

Ao analisar os resultados obtidos, pode-se notar que a mudança na regra de parada tornou o algoritmo mais estável. A exploração de novos caminhos de maior custo, principal dificuldade para se obter a convergência do algoritmo inicial, foi interrompida.

A introdução do reforço negativo trouxe melhorias ao algoritmo, conforme pode ser notado na Figura 5.12(b). Houve um aumento na rapidez de convergência, promovido pelo aumento da convergência da busca realizada por uma única formiga do algoritmo.

Além desta melhora da formiga, foram analisados os efeitos da inclusão do algoritmo cultural proposto no item (3.3.5). Para esta análise, deve-se, inicialmente, levar em conta os seguintes aspectos:

- o objetivo desta análise é mostrar a capacitação de cada heurística adicionada em aumentar a convergência com o mesmo número de agentes. Assim, embora

o algoritmo cultural previsto leve em conta um número de formigas e de colônias, o somatório do número de formigas do algoritmo não pode ser superior ao número de formigas já analisado (22 agentes). Para tal, tomou-se 3 colônias, com 2 formigas, sendo executadas 3 vezes (total de 18 execuções).

- é importante notar que as colônias são independentes, excluindo-se o espaço de crenças comum. Existem feromônios diferentes para cada formiga. Quando se adiciona o algoritmo cultural, é observado um comportamento de cooperação entre as colônias para a minimização do espaço de busca. Assim, para a análise do algoritmo cultural, deve-se levar em conta que formigas de diferentes colônias podem não estar contidas em uma mesma série de dados.

Os resultados obtidos através do algoritmo cultural são mostrados na Figura 5.13. Os dados relacionados são apresentados na Tabela 5.6. Para a execução, adotam-se as constantes $c\_min = f\_min = 2$.

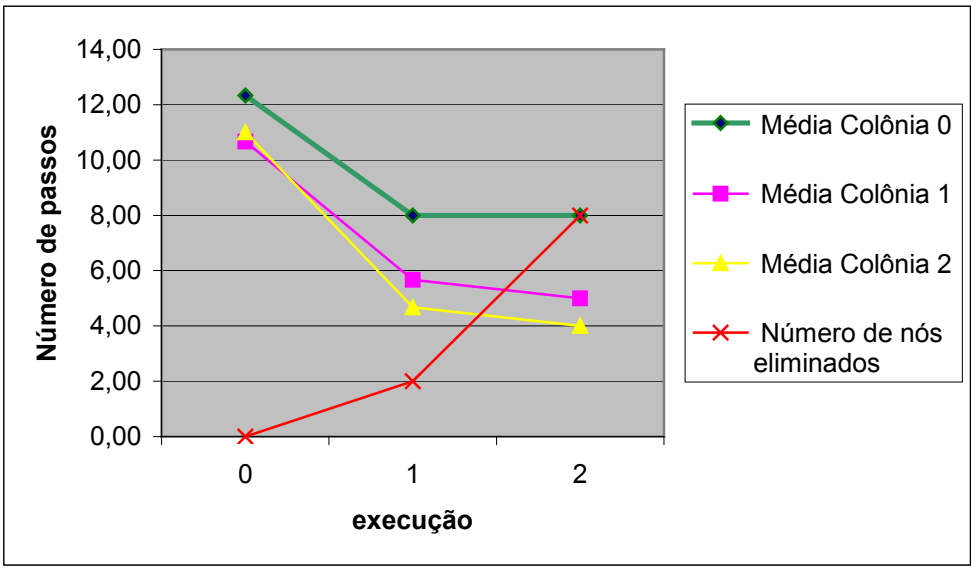


Figura 5.13 Resultados obtidos com simulação através da inclusão das heurísticas do algorítmo cultural mostradas no item 5.5.

Introdução do algoritmo cultural	Execução 0	Execução 1	Execução 2
Média de movimentos da Colônia 0	12,33	8,00	8,00
Média de movimentos da Colônia 1	10,67	5,67	5,00
Média de movimentos da Colônia 2	11,00	4,67	4,00
Número de nós eliminados	0	2	8

Tabela 5.6 Séries de dados referentes à figura 5.12.

Notou-se que a convergência encontrada foi visivelmente melhor entre as execuções das colônias quando heurísticas do algoritmo cultural foram incluídas. É observado também que o número de nós eliminados no passo final foi expressivo, sendo esta alteração no espaço de crenças não aproveitado pelo algoritmo, pois não foram criados mais agentes com este espaço de crenças reduzido. Em vista disso, uma nova análise mantendo o número de formigas e colônias foi realizada, extendendo o número de execuções para 5, e mudando o número de formigas por colônia para 3. O resultado obtido é mostrado na Figura 5.14 e na Tabela 5.7, respectivamente.

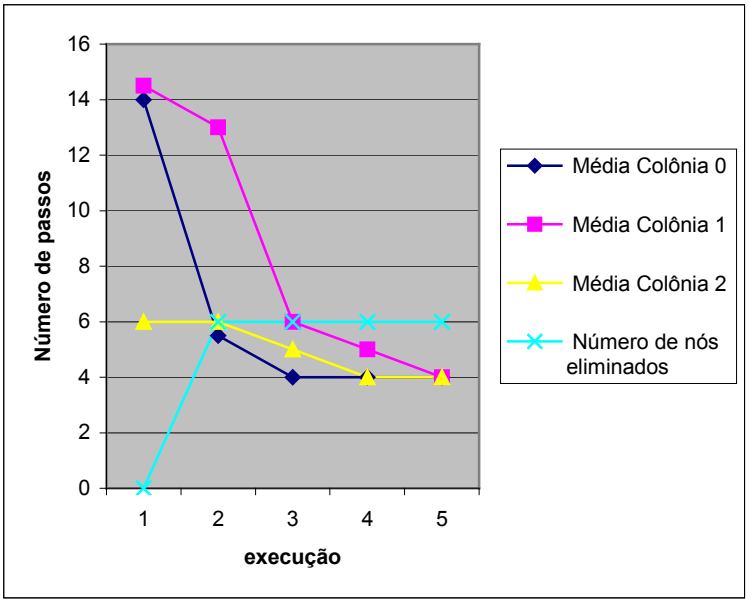


Figura 5.14 – Uma nova análise obtida com simulação através da inclusão das heurísticas do algorítmo cultural.

Execução	Média de movimentos da Colônia 0	Média de movimentos da Colônia 1	Média de movimentos da Colônia 2	Número de nós eliminados
0	14	14,5	6	0
1	5,5	13	6	6
2	4	6	5	6
3	4	5	4	6
4	4	4	4	6

Tabela 5.7 Séries de dados referentes à Figura 5.13.

Através da comparação dos resultados mostrados nas Figuras 5.12, 5.13 e 5.14 e nas Tabelas 5.5, 5.6 e 5.7, pode-se perceber que existe uma forte relação entre a eficiência do algoritmo cultural e a composição da população de agentes. Para que seja possível verificar seus efeitos, a seguir são trabalhados mapas de diferentes topologias. O próximo mapa de testes é mostrado na Figura 5.15. Neste mapa, a formiga têm seu movimento originado no nó “0” e possui como objetivo passar nos nós “9”, “10” e “14”.

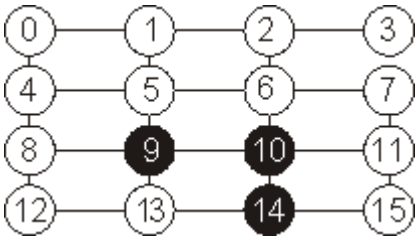


Figura 5.15 Segundo campo de testes analisado através do algoritmo cultural.

Inicialmente, foram executadas 10 vezes o algoritmo com 5 colônias simultâneas, cada uma com 5 formigas (total de 250 agentes no total, 25 por execução). A eliminação de um nó ocorria quando todas as formigas de todas as colônias não o usavam. Os resultados são mostrados na Figura 5.16 e Tabela 5.4.

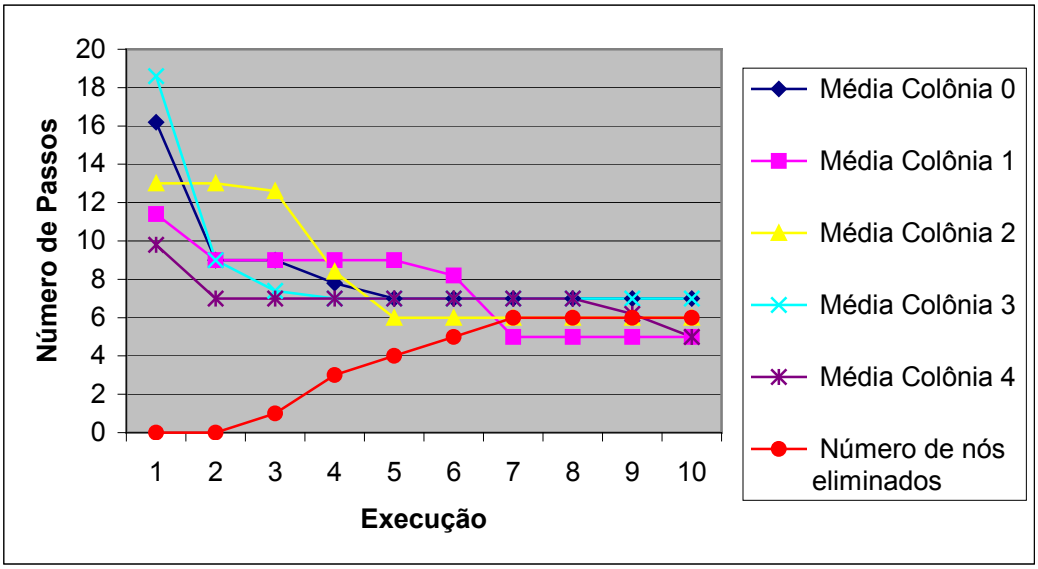


Figura 5.16 Resultados da aplicação de heurísticas de algoritmo cultural no mapa 3 através da inclusão das heurísticas do algoritmo cultural utilizando o campo de teste mostrando na figura 3.28 e *c_min*=*f_min*=5.

Execução	Média de movimentos da Colônia 0	Média de movimentos da Colônia 1	Média de movimentos da Colônia 2	Média de movimentos da Colônia 3	Média de movimentos da Colônia 4	Número de nós eliminados
0	16,2	11,4	13	18,6	9,8	0
1	9	9	13	9	7	0
2	9	9	12,6	7,4	7	1
3	7,8	9	8,4	7	7	3
4	7	9	6	7	7	4
5	7	8,2	6	7	7	5
6	7	5	6	7	7	6
7	7	5	6	7	7	6
8	7	5	6	7	6,2	6
9	7	5	6	7	5	6

Tabela 5.8 Séries de dados referentes à Figura 3.28.

Para fins de comparação, os mesmos 250 agentes foram executados, variando apenas as constantes *c_min* e *f_min* para o valor 3. Desta forma, a eliminação de um nó será realizada quando pelo menos três colônias possuírem 3 formigas que não utilizem o mesmo em seus caminhos. O resultado é mostrado na Figura 5.17 e na Tabela 5.9.

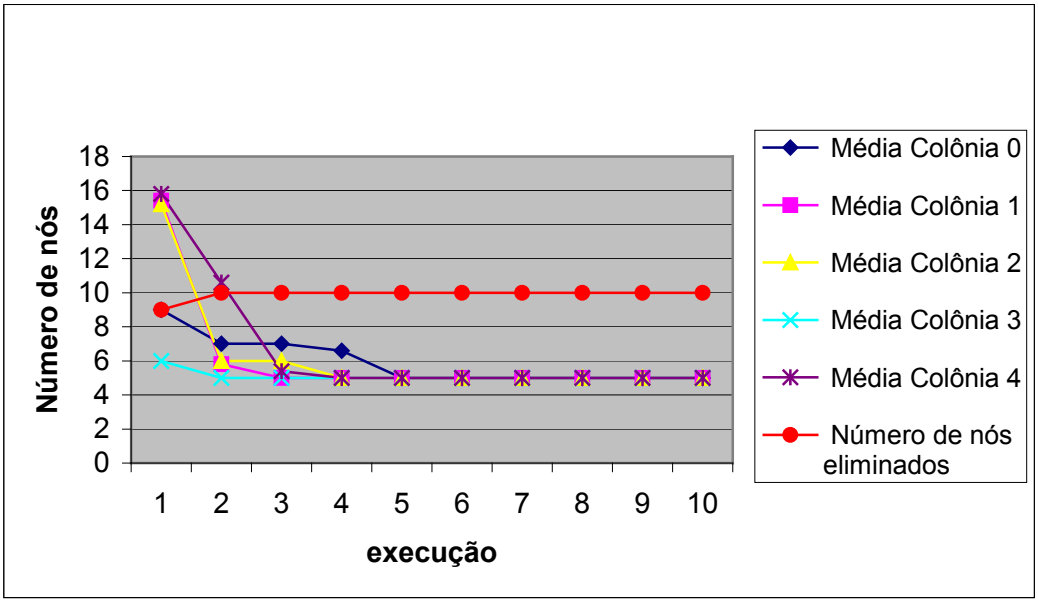


Figura 5.17 Resultado da aplicação das heurísticas com parâmetros modificados de algoritmo cultural no mapa 3 com *c_min*=*f_min*=3.

Execução	Média de movimentos da Colônia 0	Média de movimentos da Colônia 1	Média de movimentos da Colônia 2	Média de movimentos da Colônia 3	Média de movimentos da Colônia 4	Número de nós eliminados
0	9	15,4	15,2	6	15,8	9
1	7	5,8	6	5	10,6	10
2	7	5	6	5	5,4	10
3	6,6	5	5	5	5	10
4	5	5	5	5	5	10
5	5	5	5	5	5	10
6	5	5	5	5	5	10
7	5	5	5	5	5	10
8	5	5	5	5	5	10
9	5	5	5	5	5	10

Tabela 5.9 Séries de dados referentes à Figura 3.28 e *c_min*=*f_min*=3.

Ao confrontar os dados apresentados nas Figuras 5.13, 5.14 e 5.16, pode-se perceber que os parâmetros *c_min* e *f_min* influenciam na rapidez de convergência do algoritmo. Pelo que se pode constatar ao realizar estas duas comparações, a convergência foi máxima quando *c_min* e *f_min* eram o valor mínimo permitido pelas condições de contorno indicados no item 3.3.5.

Com isso, pode-se concluir que, para o problema analisado, a convergência máxima existe quando se tem a remoção de um nó do espaço de crenças possível com o menor número de formigas e colônias.

6. Conclusões e trabalhos futuros

A importância de um planejamento autônomo de rotas para a automação de robôs de serviço mostrada neste trabalho já foi apontada em outros trabalhos.

Esta pesquisa apresenta quatro conjuntos de heurísticas que, adicionadas ao algoritmo apresentado por Dorigo *et al.* (1996), buscam resolver partes do problema do planejamento de rotas para robôs de inspeção: um primeiro, onde a lista tabu é removida, um segundo, onde temos um reforço negarivo referente à memória de cada agente individual, um terceiro onde alterou-se o critério de parada para aumentar a influência de resultados melhores encontrados durante a execução, e o quarto que combina todas os algoritmos anteriores com o algoritmo cultural.

Os resultados obtidos ao se aplicar variações do algoritmo de otimização por colônia de formigas, mostraram-se promissores, sendo que dois problemas não cobertos pelo algoritmo original – a existência de um cabo umbilical e a necessidade de se passar várias vezes pelo mesmo ponto – foram resolvidos através das heurísticas adicionais propostas.

O ACS implementado confirmou as características citadas por Dorigo *et al.* (1996), no que diz respeito à robustez e à versatilidade do mesmo. A adição das mencionadas heurísticas permitiu uma melhor convergência do problema de planejamento de rotas.

Parte das heurísticas aqui apresentadas – sejam elas as definidas por Dorigo *et al.* (1996), que são relacionadas com o reforço negativo ou com as derivadas da inclusão do conceito de algoritmos culturais – possuem uma característica em comum: são compostas de um conjunto de parâmetros cujo valor ótimo varia de acordo com o cenário onde o algoritmo é executado. Esta relação foi determinada neste trabalho apenas de forma empírica.

Observa-se a importância do estudo das relações entre estes parâmetros e a rapidez de convergência do algoritmo. Espera-se, em trabalhos futuros, abordar tal tema através do uso de técnicas evolutivas, permitindo que, desta forma, não apenas o algoritmo se adapte ao meio, mas também os valores dos parâmetros possuam este comportamento adaptativo.

Deve-se frisar que a contribuição deste trabalho consistiu em combinações inéditas de heurísticas de ação restrita ao indivíduo (o reforço negativo que ocorre em função da memória de cada agente), e abrangendo todo o conjunto de agentes (no caso do algoritmo cultural, que possibilita uma segunda forma de interação entre os agentes, modificando o espaço de buscas global de acordo com as experiências individuais compartilhadas).

As heurísticas de ação restrita ao agente, se mostraram eficazes como forma de substituir a lista tabu em aplicações que depende de várias passagens em um mesmo ponto. Sua operação, em função da memória individual, permite que o agente busque novas soluções individuais ao problema, evitando assim uma interferência indesejada na busca individual de cada agente.

Ao adicionar o algoritmo cultural como heurística que permita a modificação do espaço de buscas de todos os agentes, adicionou-se uma segunda forma de cooperação. Esta cooperação, se mostrou eficaz para o aumento da convergência do algoritmo, ao permitir que todo o conjunto de agentes defina o espaço de buscas a ser considerado, descartando as regiões de menor chance de uso.

Por fim, nota-se que tais combinações, entretanto, ainda deixaram espaço para a inserção de várias outras técnicas de inteligência artificial amplamente encontrados na literatura científica. Os resultados, embora inicialmente previstos para um universo pequeno de dispositivos, podem ser aplicados em muitos outros problemas de otimização combinatórios.

Referências bibliográficas

ALAMI R.; CHATILA R.; FLEURY, S.; GHALLAB, M.; INGRAND, F. (1998). International Journal of Robotics Research, Edição Especial de "*Integrated Architectures for Robot Control and Programming*", Vol. 17, No. 4, Abril. pp. 315-337.

ALAMI, R.; INGRAND, F.; PY, F. (2002). Dependability Issues in a Robot Control Architecture. *2nd IARP/IEEE-RAS Joint Workshop on Technical Challenge for Dependable Robots in Human Environments*, LAAS-CNRS, Toulouse, France, 7-8 de outubro.

APPLEGATE, D.; BIXBY, R.; COOK, W. (1998). On the Solution of Travelling Salesman Problems. *Documenta Mathematica, Extra Volume Proceedings ICM III*, pp. 645-656. 1998.

BEVILACQUA, R..; CASTAÑO, M. C. (1999). Estratégias Adaptativas em Sociedades de Formigas. *SBA Controle & Automação*, vol. 10, pp. 167-175.

BIESZCZAD, A.; PAGUREK, B; WHITE, T. (1998). Mobile Agents for Network Management. *IEEE Communications Surveys*, vol. 1, no. 1. Disponível em citeseer.ist.psu.edu/article/bieszczad98mobile.html.

BLUM, C. ROLI, A.(2003). Metaheuristics in Combinatorial Optimization: Overview and Conceptual Comparison. *ACM Computing Surveys*, vol. 35, no. 3, pp. 268-308. Setembro de 2003.

BOCTOR, F. F.; LAPORTE, G.; RENAUD, J. (2003). Heuristics for the Traveling Purchaser Problem. *Computers & Operations Research*, vol. 30, no. 4, pp. 491-504.

BORYCZKA, M.; WIECZOREK, W. (2003). Solving Approximation Problems Using Ant Colony Programming, *Artificial Intelligence Methods*, Polônia, 5-7 de novembro.

BONABEAU, E.; HENAUX, F.; GERIN, S.; SNYERS, D.; KUNTZ, P.; THERAULAZ, G. (1998). Routing in Telecommunications Networks with "Smart" Ant-like Agents. *Intelligent Agents for Telecommunications Applications*. Second International Workshop IATA '98. Vol. 1437. Julho de 1998. Disponível em: http://citeseer.ist.psu.edu/bonabeau98routing.html

BONABEAU, E.; THÉRAULAZ, G. (1998). Swarm Smarts. *Scientific American*, pp. 82-90, Março. Disponível em http://dsp.jpl.nasa.gov/members/payman/swarm/.

BONABEAU, E.; DORIGO, M.; THERAULAZ, G. (2000). Inspiration for Optimization from Social Insect Behaviour. *Nature*, vol. 406, pp. 39-42.

BOTEE, H. M.; BONABEAU, E.(1999). Evolving Ant Colony Optimization. *Advances in Complex Systems*, vol. 1, no. 2/3, pp. 149-159.

BOURJOT, C.; CHEVRIER, V.; THOMAS, V.(2003). A new Swarm Mechanism Based on Social Spiders Colonies: From Web Weaving to Region Detection. *Web Intelligence and Agent Systems: An International Journal I*. IOS Press. pp. 47-64.

BROOKS, R. A.(1991). Intelligence Without Reason. *Proceedings of the 12th International Joint Conference on Artificial Intelligence ({IJCAI}-91)*. Sydney, Australia. pp. 569-595. Agosto.

BYRNE, R. H.; ADKINS, D. R.; ESKRIDGE, S. E.; HARRINGTON, J. J.; HELLER, E. J.; HURTADO, J. E. (2002). Miniature Mobile Robots for Plume Tracking and Source Localization Research. *Journal of Micromechatronics*, vol. 1, no. 3, pp. 253-261.

CHOI, H. R.; RYEW, S. M. (2002); Robotic System with Active Steering Capability for Internal Inspection of Urban Gas Pipelines. *Mechatronics*, no. 12, pp. 713-736.

CHOSET, H.; KONUKSVEN, I.; RIZZI, A. (1996). Sensor Based Planning: A Control Law for Generating the Generalized Voronoi Graph. *Proc. IEEE Int. Advanced Robotics*, Washington, EUA. Disponível em: http://citeseer.ist.psu.edu/article/choset96sensor.html

CHOSET, H.; NAGATAMI, K. (2001). Topological simultaneous localization and mapping (SLAM): toward exact localization without explicit localization. *IEEE Transactions on Robotics and Automation*, vol. 17, no. 2, pp.125-137. Disponível em http://voronoi.sbp.ri.cmu.edu/papers.english.html.

COELHO, L. S.; COELHO, A. A. R.(1999); Algoritmos Evolutivos em Identificação e Controle de Processos: uma Visão Integrada e Perspectivas. *Revista Controle e Automação*. vol. 10. no. 01. pp 13-30.

COELLO, C. A. C. (2002); Theoretical and Numerical Constraint-handling Techniques used with Evolutionary Algorithms: a Survey of the State of the Art. *Computer Methods in Applied Mechanics and Engineering*. no. 81, pp. 1245-1286.

COELLO, C. A. C. C; BECERRA, R. L.(2003); A Cultural Algorithm for Constrained Optimization Second Mexican International Conference on Artificial Intelligence. pp. 98-117. Merida, México.

COHEN, W. (1996). Adaptive Mapping and Navigation by Teams of Simple Robots. Robotics and Autonomous Systems. no. 18, pp. 411-434.

COSTA, V. M. S. (1998). Integração de Comportamentos Visuais para Robótica Móvel. Dissertação de mestrado. Universidade Técnica de Lisboa, Portugal. Disponível em http://vislab.isr.ist.utl.pt/theses.html

COURY, D. V. (2002). Algoritmos Genéticos: Uma Nova Abordagem Para a Resolução de Problemas em Sistemas Elétricos de Potência, *Edital Universal CNPq 01/2002*. Projeto de Pesquisa.

CUNHA, C.B; BONASSER, U.O. e ABRAHÃO, F.T.M. (2002) "Experimentos Computacionais com Heurísticas de Melhorias para o Problema do Caixeiro Viajante". In: *Setti, J.R.A. e Santos, E.M., eds (2002) Panorama Nacional de Pesquisa em Transportes* – Anais do XVI ANPET, ANPET, Rio de Janeiro, RJ, Vol. 1, pp.105-117.

DATTERI, E.; TETI, G.; LASCHI, C.; TAMBURRINI, G.; DARIO, P.; GUGLIELMELLI, E. (2003) Expected perception in robots: a biologically driven perception-action scheme. *ICAR 2003, 11th International Conference on Advanced Robotics*, pp.1405-1410.

DEB, K.(1999); An Introduction to Genetic Algorithms. Kanpur Genetic Algorithm Lab. http://www.iitk.ac.in/kangal/pub.htm. Acesso em 16/04/2004.

DENEUBOURG, J. L.; PASTEELS, J. M.; VERHAEGHE, J. C. (1983). Probabilistic behaviour in ants: a strategy of errors? *Journal of Theoretical Biology*, no. 105, pp. 259-271.

DIXON, K. R.; STRAND, M.; KHOSLA, P. K.(2002). Predictive Robot Programming. *IEEE/RSJ International Conference on Intelligent Robots and Systems*, EPFL, Suíça. Disponível em http://chernoybl.ices.cmu.edu:8080/papers/.

DORIGO, M.; MANIEZZO, V.; COLORNI, A. (1996). The Ant System: Optimization by a Colony of Cooperating Agents. *IEEE Transactions on Systems, Man and Cybernetics* - Part B, vol. 26, no. 1, pp. 1-13.

DORIGO, M.; GAMBARDELA, L. M.(1997) Ant Colonies for the Travelling Salesman Problem. *BioSystems*, vol. 43, pp. 73-81.

DORIGO, M.; TRIANNI, V.; SAHIN, E.; GROB, R.; LABELLA, T. H.; BALDASSARE, G.; NOLFI, S.; DENEUBOURG, J. L.; MONDADA, F.; FLOREANO, D.; GAMBARDELLA, L. M. (2004) Evolving Self-Organizing Behaviors for a Swarm-Bot. *Autonomous Robots*, no. 17, pp. 223-245.

FIRA (2005). Federation of International Robot-soccer Association. Página HTML disponível em http://www.fira.net/media/photo/read.html?indexnum=342&page=1&keyword=&depth1=1&depth2=&depth3= Acesso em 20 de fevereiro de 2005.

FOX, S. (2004); Swarm Stupidity. IEE Manufacturing Engineer. Dezembro(2003)-Janeiro (2004), pp. 26-29.

FRANKLIN, S.; GRAESSER, A. (1996). Is it an Agent or Just a Program? A Taxonomy for Autonomous Agents. *12.a European Conference on AI*. Berlim, Alemanha. Disponível em http://www.msci.memphis.edu/%7Efranklin/AgentProg.html

FULLER, J. L.(1999); An Introduction to Robotics. Prentice Hall, Upper Saddle River, NT.

GAGE, D. W. (1995); Many-Robot MCM Search Systems. *Autonomous Vehicles in Mine Countermeasures Symposium*. pp. 9.56-9.64. Abril de 1995.

GASPAR, J; WINTERS, N.; GROSSMANN, E.; VITOR, J. S. (2004). Toward Robot Perception using Omnidirectional Vision, *Innovations in Machine Intelligence and Robot Perception* disponível em http://www.lkl.ac.uk/niall/book_odv.pdf

GOOGLE (2004). *Google*. Página HTML disponível em http://www.google.com em 7 de dezembro de 2004.

HAGRAS, H.; CALLAGHAN, V.; COLLEY, M. (2001) Outdoor Mobile Robots Learning and Adaptation. *IEEE Robotics & Automation Magazine*. no. 53. pp. 53-69. Setembro.

HELSGAUN, K.(1998). An Effective Implementation of the Lin-Kernighan Travelling Salesman Heuristic. *DATALOGISKE SKRIFTER (Writings on Computer Science)*, No. 81. pp 106-130.

IACOBAN, R.; REYNOLDS, R. G.; BREWSTER, J. (2003). Cultural Swarms: Assessing the Impact of Culture on Social Interaction and Problem Solving. *IEEE Swarm Intelligence Symposium*. pp 205-211. Abril.

INSTITUTO PARA A ROBÓTICA PRÁTICA (2005). BotBall – Touch Sensors. Página HTML Disponível em http://www.kipr.org/curriculum/touch.html. Acesso em 15 de fevereiro de 2005.

INUKTUM (2005). Minitrac robotic propulsion units. Página HTML disponível em http://www.inuktun.com/minitracs.htm. Acesso em 20 de fevereiro de 2005.

JIANXU, M.; XIANG, L.; JIANHUA, M.; MINGDONG, L.; PEISUN, M. (2000). Microbionic and Peristaltic Robots in a Pipe. *Chinese Science Bulletin*. Vol. 45, No. 11, Junho.

JIN, X.; REYNOLDS, R. G. (1999). Using Knowledge-Based System with Hierarchical Architecture to Guide the Search of Evolutionary Computation. *IEE International Conference on Tools with Artificial Intelligence (ICTAI'1999)*, pp 29-36. Chicago, Illinois, EUA.

JULY, 2002, Sant`Angelo d`Ischia, Italy. *Springer Tracts in Advanced Robotics 5*, pp. 297-306.

KANOH, H.; NAKAMURA, T. (2000). Knowledge Based Genetic Algorithm for Dynamic Route Selection. *IEE Fourth International Conference on Knowledge Based Intelligent Engineering Systems and Allied Technologies (KES'2000)*, pp. 616-619. Brighton, Reino Unido.

KARLSON, P. (2002). Simulated Annealing Applied to the Travelling Salesman Problem. Technical Report, Universidade de Upsala, Canadá.

KAZADI, S. (2002). Swarm Engineering. *Tese de Doutorado em Filosofia*. Instituto de Tecnologia da Califórnia. Pasadena, Califórnia, EUA.

KELLY, K. (1995). Out of Control: The New Biology of Machines, Social Systems and the Economic World. *Perseus Publishing*. Nova York, EUA.

KIRKPATRICK, S.; GELATT, C. D.; VECCHI, M. P. (1983). Optimization by Simulated Annealing. *Science*. vol. 220, no. 4598, pp. 671-680.

KITA, N.; DAVISON, A.; KUNIYOSHI, Y.; HARA, I.; MATSUI, T.; ROUGEAUX, S.; HORI, T.; HIRAI, S. (1999). Mobile Sensing Robots for Nuclear Power Plant Inspection. *Advanced Robotics*, vol. 13. no. 3. pp. 355-356.

KLJAJIC, M., BRESKVAR, U., BERNIK, I. (2002). Production Planning Using a Simulation Model and Genetic Algorithms. *IASTED International Conference Proceedings*. Marina Del Rey, California, EUA. pp 54-58.

KOENING, S. (1999). Robot Localization and Exploration with Agent-Centered Search. *International Joint Conference on Artificial Intelligence*. Disponível em citeseer.ist.psu.edu/207440.html. Acesso em 27 de setembro de 2003.

LAPORTE, G. (2000) Classical and modern heuristics for the vehicle routing problem. *International Workshop on Vehicle Routing*. Skodsborg, Dinamarca. Agosto. Disponível em http://www.imm.dtu.dk/conferences/route2000/ Acesso em 27 de setembro de 2003.

LAVINE, B. K.; DAVIDSON, C.; MEER, R. K. V.; LAHAV, S.; SOROKER, V.; HEFETZ, A. (2003). Genetic Algorithms for Deciphering the Complex Chemosensory Code of Social Insects. *Chemiometrics and Intelligent Laboratory Systems*. no. 66. pp. 51-62.

LERMAN, K.; GALSTYAN, A.; MARTINOLI, A.; IJSPEERT, A. (2001). A Macroscopic Analytical Model of Collaboration in Distributed Robotic Systems. *Artificial Life*, no. 7, pp. 375-393.

LIN, S.; KERNIGHAN, B. W. (1973). An Effective Heuristic Algorithm for the Travelling Salesman Problem. *Operational Research*, vol. 21, no. 2, pp. 498-516, 1973.

LOCKHEED, M. (1997). Multisensor Inspection & Caracterization Robot for Small Pipes (MICROSPI). *Technology Development Data Sheet*. Federal Energy Technology Center.

LOW, K. H.; LEOW, W. K., ANG, M. H. (2002). A Hybrid Mobile Robot Architecture with Integrated Planning and Control. *I AAMAS' 02*, pp. 219-226. Bolonha, Itália.

LYNXMOTION (2005). Sensors. Página HTML disponível em http://www.lynxmotion.com/Category.aspx?CategoryID=8. Acesso em 15 de fevereiro de 2005.

MARTINOLI, A.; EASTON, K. (2002). Modeling Swarm Robotic Systems. In B. Siciliano and P. Dario, editors, *Proc. of the Eight Int. Symp. on Experimental Robotics ISER-02*,
JONES, J. L.; FLYNN, A. M.; SEIGER, B. A. (1998) Mobile Robots: Inspiration to Implementation. Ed. AK Peters Ltd; 2ª. edição.

MERZ, P.; FREISLEBEN, B. (2000). Fitness Landscapes, Memetic Algorithms and Greedy Operators for Graph Bi-Partitioning. *IEEE Transactions on Evolutionary Computation*, vol. 4, no. 4, 2000.

Metropolitan Sewer District: What Happens When You Flush? Página HTML disponível em http://www.msdlouky.org/programs/whenuflush/maintain.htm, acesso em 30 de janeiro de 2003.

MEULEAU, N.; DORIGO, M. (2002). Ant Colony Optimization and Stochastic Gradient Descent, *Artificial Life* , no. 8, pp. 103-121.

McKERROW, J. P. (1991). Introduction to Robotics. Livro, ISBN ISBN 0 201 18240 8. 811 páginas.

MORANDI, M. B.; NAZEMI, E.; An Approach to Genetic Algorithm Path Planning Used in RobocupRescue. Disponível em http://www.sbcee.net/rescueteams/sbceres2003.html Acesso em 15 de agosto de 2003.

MILLS, P.; TSANG, E.; ZHANG, Q.; Ford, J. (2004); A Survey of AI-based Meta-heuristics for Dealing with Local Optima in Local Search. Technical Report, disponível em http://cswww.essex.ac.uk/technical-reports/index.htm Acesso em janeiro de 2005.

MONTEIRO, S. T.; RIBEIRO, C. H. C.; Desempenho de Algoritmos de Aprendizagem por Reforço sob Condições de Ambiguidade Sensorial em Robótica Móvel. *SBA Controle & Automação*, vol. 15 no. 3, 2004.

MORAWEK, R. (2003); Path Finding for Mobile Agents or the Travelling Salesman Problem under Incomplete Information. *Tese de doutorado*, Technischen Universität Wien. pp. 65-83.

NASA (2005); sojourner.jpg. Imagem disponível em http://nssdc.gsfc.nasa.gov/image/spacecraft/sojourner.jpg. Acesso em 20 de fevereiro de 2005.

NEHMZOW, U. (2001). Mobile Robotics: Research, Applications and Challenges. *Proc. Future Trends in Robotics*, Institution of Mechanical Engineers, London.

New York Gas Group (2001). EXPLORER: A Long Range Untethered Live Gasline Inspection Robot System. *NYGAS Technology Brief*, Abril/2001.

Oldenburg, Universidade de; Marine Physics. Página HTML disponível em http://las.physik.uni-oldenburg.de/general/activities.html em 27 de setembro de 2003.

OJALA, J. (1998). Adaptative Behavior with Protozoa-Inspired Dynamics. *Biological Cybernetics* no. 79, pp. 403-411.

OSTROWSKI, D. A.; TASSIER, T.; EVERSON, M. P.; REYNOLDS, R. G. (2002). Using Cultural Algorithms to Evolve Strategies in Agent-Based Models. *Proceedings of the 2002 Congress on Evolutionary Computation*, pp. 741-746. Honolulu, HI, EUA.

OZAKI, K.; YOKOTA, K.; MATSUMOTO, A.; ASAMA, H. (1999). Communication System for Cooperative Mobile Robots – Implementation of Communication Among Soccer Robots. *Advanced Robotics*. vol. 13, no. 3, pp 287-288.

PANETTA, P.D.; DIAZ, A.A.; PAPPAS, R.A.; TAYLOR, T.T.; FRANCINI, R.B.; JOHNSON, K.I.; *Mechanical Damage Characterization in Pipelines*. U.S. Department of Energy. Outubro/2001. Disponível em http://www.netl.doe.gov/scng/trans-dist/ngdel/tech-status.html em 27 de setembro de 2003.

PARPINELLI, R.S.; LOPES, H.S.; FREITAS, A.A.(2002). An Ant Colony Algorithm for Classification Rule Discovery. In: H. Abbass, R. Sarker, C. Newton. (Eds.) *Data Mining: a Heuristic Approach*, pp. 191-208. London: Idea Group Publishing.

PARKER, C. A. C.; ZHANG, H.; KUBE, C. R. (2003). Blind Bulldozing: Multiple Robot Nest Construction. *Proc. IEEE/RSJ International Conference on Robotics and Intelligent Systems*, Las Vegas, USA, Outubro 27-31.

P.A.R.T.S. (2005). Accelerometer Kit. Portland Area Robotics Society. Página HTML disponível em http://www.junun.org/MarkIII/Info.jsp?item=6#. Acesso em 20 de fevereiro de 2005.

Paul Hayward Associates. (2003). Paul Hayward Associates. Página HTML disponível em http://www.webcare.co.uk/pha/pha1.htm em 27 de setembro de 2003

PETTINARO, G. C.; KWEE, I. W.; GAMBARDELLA, L. M.; MONDADA, F.; FLOREANO, D.; NOLFI, S.; DENEUBOURG, J. L.; DORIGO, M. (2002). Swarm Robotics: A Different Approach to Service Robots. *Proc. 33 International Symposium on Robotics*. 7-11 de Outubro de 2002. Disponível em http://gral.ip.rm.cnr.it/nolfi/nolfipub.html

PFEIFER, R.; SCHEIER, C. (1994). From perception to action: the right direction ? *Proc. From Perception to Action Conference*, pp. 1-11. IEEE Computer Society Press, 1994. Disponível em http://citeseer.ist.psu.edu/pfeifer94from.html.

QUEIROZ, L. R.; BERGERMAN, M.; MACHADO, R. C.; BUENO, S. S.; ELFES, A. (1998). Educação a Distância em Robótica e Visão Computacional. *Revista Brasileira de Informática na Educação*, no. 3. Disponível em http://www.inf.ufsc.br/sbc-ie/revista/nr3/

RANGANATHA, A.; KOEING, S (2003). A Reactive Robot Architecture with Planning on Demand; *Intelligent Robots and Systems for Human Security, Health, and Prosperity*. Outubro de 2003. Disponível em http://citeseer.ist.psu.edu/682586.html

RAVIKUMAR, C. N., GOWDA, K. C.; NAGABHUSHAN, P. (1998). An Efficient Technique to Solve TSP and its Extension to Spatially Spread Vertices. *Current Science*, vol. 75, no. 10, pp. 1046-1052.

RESENDE, M. G. C.; VELARDE, J. L. G. (2003). GRASP: Greedy Randomized Adaptative Search Procedures. *Revista Iberoamericana de Inteligência Artificial*. no. 9, pp 61-76.

REYNOLDS, R. G.; CHUNG, C. J. (1997). Knowledge-Based Self Adaptation in Evolutionary Search, *Proceedings of 1997 IEEE International Conference on Artificial Intelligence Tools*, Newport Beach, Indianapolis, EUA

REYNOLDS, R. G.; Zhu, S. (2001). Knowledge-based Function Optimization using Fuzzy Cultural Algorithms with Evolutionary Programming. *IEEE Transactions on Systems, Man and Cybernetics* – Part B: Cybernetics. vol. 31, no. 1, pp. 19-31.

ROBOT BOOKS (2005). Robot Books.com – Robot Kits, Robotics, and Toy Robots. Página HTML disponível em http://www.robotbooks.com/. Acesso em 20 de fevereiro de 2005.

ROBOPROBLE TECNOLOGIES INC. *List of Robot Crawlers*. Página HTML disponível em http://www.roboprobe.com/CATALOGS/C38-LIST_ROBOTS_AND_ROVS.HTML em 27 de setembro de 2003.

ROMAO, W.; FREITAS, A. A.; PACHECO, R. S. (2002). A Genetic Algorithm for Discovering Interesting Fuzzy Prediction Rules: Applications to Science and Technology Data *Proceedings of Genetic and Evolutionary Computation Conference (GECCO-2002).* São Francisco, EUA. Pp 1188-1195 Julho.

SANDI, F. A.; HEMERLY, E. M.; LAGES, W. F. (1998). Sistema de Navegação e Guiagem de Robôs Móveis e Autônomos, *Revista SBA Controle e Automação,* vol. 9, no. 3, pp. 107-118.

SCHOONDERWOERD, R.; HOLLAND, O.; BRUTEN, J.; ROTHKRANTZ, L. (1996). Ant-Based Load Balancing in Telecommunications Networks. *Adaptive Behavior*. no. 2, pp. 169-207.

SENSOR RESEARCH (2005). Sensors Research Consulting, Inc. Página html disponível em http://www.sensors-research.com/, acesso em 06/10/2005.

SILVER, E. A. (2002). An Overview of Heuristic Solution Methods. *Working paper*. Haskayne School of Business, University of Calgary, October, 2002.

SIM, K. M.; SUN, W. H.(2003) Ant Colony Optimization for Routing and Load-Balancing: Survey and New Directions. *IEEE Transactions on Systems, Man and Cybernetics – Part A: Systems and Humans*, vol. 33, no. 5, pp. 560-572.

Sony (2005). What is AIBO. Página HTML disponível em http://www.aibo-europe.com/1_1_3_ers7_1.asp Acesso em 20 de fevereiro de 2005.

SRI International. *Industrial Products*. Página HTML disponível em http://www.sri.com/ipet/indusprds.html#it em 27 de setembro de 2003

STAGER, B. (2001). The Solving of Real World Problems using Evolutionary Algorithms. *Undergraduate Thesis*. University of Queensland. Outubro.

SIEGWART, R.; NOURBAKHSH, I. R. (2004). Introduction to Autonomous Mobile Robots. MIT Press. Cambridge, EUA.

SUTTON, R. S. (1992). Reinforcement Learning Architectures. *Proceedings of ISKIT'92 International Symposium on Neural Information Processing*. Disponível em http://www.cs.ualberta.ca/~sutton/publications.html. Acesso em 27 de setembro de 2003.

TALBI, E. G.; ROUX, O. ; FONLUPT, C., ROBILLARD, D. (2001). Parallel Ant Colonies for the Quadratic Assigment Problem. *Future Generation Computer Systems*, no. 17, pp. 441-449.

TAHERI, S. A.; CALVA, G.(2001). Imitating the Human Imune System Capabilities for Multi-Agent Federation Formation, *Proceeding of the IEEE Internations Symposium on Intelligent Control*. Cidade do México, México. pp 25-30.

TAVARES NETO, R. F.; COELHO, L. S. (2003). Proposta de Planejamento de Rotas para Robôs de Inspeção Usando um Algoritmo de Colônia de Formigas. VI SBAI – *Simpósito Brasileiro de Automação Inteligente*. Bauru, SP. Disponível em http://www.tavares.eng.br

TAVARES NETO, R. F.; COELHO, L. S. (2004). Planejamento de Rotas para Robôs de Inspeção Usando uma nova abordagem de Swarm Intelligence. I NANOBIO - Workshop em Nanotecnologia e Computação Inspirada na Biologia. Rio de Janeiro, RJ. Disponível em http://www.tavares.eng.br

TAVARES NETO, R. F.; COELHO, L. S. (2004). Planejamento de Rotas de Entrega Usando uma Abordagem de Algoritmo de Colônia de Formigas com Lista Tabu Suprimida. *XV Congresso Brasileiro de Automática*. Gramado, RS. Setembro. Disponível em http://www.tavares.eng.br

TAVARES NETO, R. F.; COELHO, L. S. (2004). Colônia de Formigas: Uma Abordagem Promissora para Aplicações de Atribuição Quadrática e Projeto de Layout. *XXIV ENEGEP – Encontro Nacional de Engenharia de Produção*. Florianópolis. Disponível em http://www.tavares.eng.br

TAVARES NETO, R. F.; COELHO, L. S. (2005). Planejamento de rotas para Robôs de Inspeção Usando um Algoritmo Híbrido de Colônia de Formigas e Algoritmo Cultural. VII SBAI – *Simpósito Brasileiro de Automação Inteligente.* São Luís. Disponível em http://www.tavares.eng.br

TAUB, H. (1984) Circuitos Digitais e Microprocessadores. *Editora McGraw-Hill do Brasil*. Pp. 239. São Paulo.

TERRA (2004). *Terra – Quem quer MAIS assina o Terra*. Página HTML disponível em http://www.terra.com.br em 7 de dezembro de 2004.

THAKOOR, S.; CHAHL, J.; SRINIVASAN, M.V.; YOUNG, L., WERBLIN, F.; HINE, B.; ZORNETZER, S. (2002) Bioinspired Engineering of Exploration Systems for NASA and DoD, *Artificial Life*, No. 8, pp. 357-369.

THOMAZ, C. E.; PACHECO, A. C.; VELLASCO, M. M. B. R. (1999). Mobile Robot Path Planning Using Genetic Algorithms. International Conference on Information Systems

Analysis and Synthesis, Image and Signal Processing. vol. 6, pp. 671-679, Alicante, Espanha.

TOMASSINI, M. (1995). Genetic Algorithms: a Survey. *Annual Reviews of Computational Physics* Vol. III, World Scientific, pp. 87-117.

ÜNSAL, C. (1993). Self Organization in Large Population of Mobile Robots. *Dissertação de mestrado*. Blacksburg, Virginia. Maio.

UOL (2004). *UOL – O Melhor Conteúdo*. Página HTML disponível em http://www.uol.com.br em 7 de dezembro de 2004.

VENUGOPAL, K. V.(2001) State-of-the-art Natural Gas Pipe Inspection. Oak Ridge National Laboratory. Novembro. Disponível em http://www.netl.doe.gov/scng/trans-dist/ngdel/tech-status.html

VIEIRA, F. C., DÓRIA Neto, A. D., e COSTA, J. A. F. (2003). An Efficient Approach To The Travelling Salesman Problem Using Self-Organizing Maps. *International Journal of Neural Systems*. Vol. 13, No. 2, pp. 59-66.

WASSON, G.; KORTENKAMP, D.; HUBER, E. (1999). Integrating Active Perception with an Autonomous Robot Architecture. *Robotics and Automation Journal*, Vol. 29, pp. 175-186, 1999.

WATTANASIN, C.; AIYAMA, Y.; KURABAYASHI, D.; OTA, J., ARAI, T. (2001). Hybrid Power Suply for Mobile Robots. *Advanced Robotics*, vol. 15, no. 6, pp. 695-710.

WHITTAKER, W.; URMSON, C.; STARITZ, P.; KENNEDY, B.; AMBROSE, R. (2000). - Robotics for Assembly, Inspection, and Maintenance of Space Macrofacilities. AIAA Space 2000 Conference and Exposition. Setembro. Long Beach, Canadá, EUA. Disponível em http://www.ri.cmu.edu/centers/sri/pubs_text.html

ZAGROS ROBOTICS (2005). Welcome to Zagros Robotics. Página HTML disponível em https://www.zagrosrobotics.com/shop/category.asp?catid=3. Acesso em 20 de fevereiro de 2005.

ZIMMER, U. R. (1995). Adaptative Approaches to Basic Mobile Robot Tasks. Tese de doutorado. Universidade de Kaiserslautem, Alemanha.

ZHONG, W.; EVANS, D. (2002). When Ants Attack: Security Issues for Stigmergic Systems. UVA CS Technical Report. Universidade de Virgínia, EUA, 2002. Disponível em http://www.cs.virginia.edu/~evans/pubs/stigmergy.html. Acesso em 27 de setembro de 2003.