RECARGA DE REATORES NUCLEARES UTILIZANDO REDES CONECTIVAS DE COLÔNIAS ARTIFICIAIS

Alan Miranda Monteiro de Lima

TESE SUBMETIDA AO CORPO DOCENTE DA COORDENAÇÃO DOS PROGRAMAS DE PÓS-GRADUAÇÃO DE ENGENHARIA DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE DOUTOR EM CIÊNCIAS EM ENGENHARIA NUCLEAR.

Aprovada por:

_____
Prof. Roberto Schirru, D.Sc.

_____
Prof. Fernando Carvalho da Silva, D.Sc.

_____
Prof. Aquilino Senra Martinez, D.Sc.

_____
Prof. Cláudio Márcio do Nascimento Abreu Pereira, D.Sc.

_____
Prof. Hermes Alves Filho, D.Sc.

_____
Dr. Antonio César Ferreira Guimarães, D.Sc.

RIO DE JANEIRO, RJ - BRASIL
JUNHO DE 2005

DE LIMA, ALAN MIRANDA MONTEIRO

Recarga de Reatores Nucleares Utilizando Redes Conectivas de Colônias Artificiais [Rio de Janeiro] 2005

X, 153 p.29,7 cm (COPPE/UFRJ, D.Sc., Engenharia Nuclear, 2005)

Tese - Universidade Federal do Rio de Janeiro, COPPE

1 – Colônias Artificiais

2 – Algoritmos Paralelos

3 – Recarga de Reatores Nucleares

I. COPPE/UFRJ II. Título (série)

Dedicatória:

À minha mãe, Nancy, que só pela palavra mãe já diz tudo.

Às traças, que certamente um dia, irão se deliciar com essas páginas suculentas de papel, abandonadas em uma estante qualquer.

À democracia por podermos expressar nossas idéias.

Ao sistema de ensino e pesquisa de nosso país, cujo continuísmo, não os deixa perceber que somos obrigados a publicar tantos artigos "Ricuperianos", que nem temos tempo para perceber seu real valor.

Ao sofrido povo brasileiro, que comemora a cada gol marcado e a cada carnaval, e talvez pela sua imbecilidade total e absoluta deixa de perceber a falta que a educação faz a um país, e apesar disto tudo ainda consegue sorrir e ser gentil. Talvez o dia em que se perceba que duas palavras são essenciais a nosso país as coisas melhorem: EDUCAÇÃO a médio e longo prazo para nossas crianças e EXECUÇÃO com confisco de bens para nossos bandidos, principalmente para os quadrilheiros do poder.

Agradecimentos:

Ao CNPq pelo apoio financeiro que possibilitou a realização desta tese;

À minha mãe Nancy, minhas tiazinhas e toda minha família, presentes e ausentes deste mundo e velhos amigos.

Ao Prof. Roberto Schirru, pela sua orientação, bom senso e boas idéias, que certamente o colocam entre as pessoas mais inteligentes que conheci, mas isso se torna pequeno se comparado com sua bondade perante à necessidade de outras pessoas.

Ao Prof. Fernando Carvalho pela sua orientação em física de reatores e pelas peladas de início de mestrado.

Aos amigos Wagner Sacco, Kelling, Vinícius, Serginho, Marquinho, João e Auroro, pelo convívio que tivemos todo esse tempo e que fazia o trabalho parecer bem mais agradável. E ao Pius, um amigo que estamos todos torcendo pela sua recuperação.

Ao Marcelo pela amizade e pela sua boa vontade em ajudar aos outros.

Ao Cláudio pela sua amizade e empolgação em tudo que faz.

Ao Carlão, uma máquina de trabalhar e grande amigo.

Ao Airton e Henrique, por "consertarem" nossos micros e pela sua amizade.

Ao Jorge Wagner pela sua amizade, bom coração e por nos fazer sentir que o nosso ego era mínimo ou nem existia.

Aos demais alunos de mestrado e doutorado, pelo clima alegre de colégio: Bárbara, Mauro, Gustavo, Anderson, Berg e Rafael que nos diverte com suas estórias de barão de Munchausen.

Aos funcionários e estagiários do LMP, pela cordialidade com que sempre nos trataram, pela amizade e pelos ensinamentos que nos passaram: Serjão, Fred, Márcio, César, Norberto, Zé Luiz, Junior, Daniel e Flamarion.

À Simone, a secretária nota 10.

Aos funcionários da ETN que sempre estavam a disposição para ajudar e nos ensinar sobre recarga: Wanderlei, Teresinha e Mandarano.

Aos amigos Márcio Dornellas e João Calixto, por sua ajuda em problemas logísticos na ETN e a Marcos Fernando que se tornou também um grande amigo.

Ao amigo Chapot, por me apresentar ao mundo nuclear e aos algoritmos evolucionários.

Resumo da Tese apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Doutor em Ciências (D.Sc.)

RECARGA DE REATORES NUCLEARES UTILIZANDO REDES CONECTIVAS DE COLÔNIAS ARTIFICIAIS

Alan Miranda Monteiro de Lima

Junho/2005

Orientadores: Roberto Schirru

Fernando Carvalho da Silva

Programa: Engenharia Nuclear

O processo de recarga de um Reator Nuclear a Água Pressurizada (RNAP) ocorre toda vez que a queima dos elementos combustíveis (EC) no núcleo do reator atinge um determinado valor em que não é mais possível manter o reator crítico produzindo energia à potência nominal.

O problema de otimização da recarga consiste em determinar o posicionamento destes EC (velhos e novos) no núcleo do reator de forma otimizada. A otimização de recargas de RNAP é um problema do tipo NP-Completo, ou seja, a dificuldade cresce exponencialmente com o número de EC no núcleo do reator. Além desta dificuldade, este problema tem características não lineares, descontinuidades, múltiplos máximos e mínimos locais no seu espaço de soluções.

No presente trabalho será apresentado um novo sistema computacional paralelo para realizar a recarga nuclear, o Redes Conectivas de Colônias Artificiais (RCCA), baseado no Sistema de Colônias de Formigas (SCF).

O SCF é um sistema baseado em Agentes Artificiais e que faz uso da técnica de Reforço de Aprendizagem. Este algoritmo foi desenvolvido inicialmente para solucionar o Problema do Caixeiro Viajante que é um problema combinatório conceitualmente similar ao da recarga nuclear.

Abstract of Thesis presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Doctor of Science (D.Sc.)

A NUCLEAR REACTOR CORE FUEL RELOAD OPTIMIZATION USING ARTIFICIAL COLONY CONNECTIVE NETWORKS

Alan Miranda Monteiro de Lima

June/2005

Advisors: Roberto Schirru

Fernando Carvalho da Silva

Department: Nuclear Engineering

A Pressurized Water Reactor core must be reloaded every time the fuel burnup reaches a level when it is not possible to sustain nominal power operation.
The nuclear core fuel reload optimization consists in finding a burned-up and fresh-fuel-assembly pattern that maximizes the number of full operational days. This problem is NP-hard, meaning that complexity grows exponentially with the number of fuel assemblies in the core. Besides that, the problem is non-linear and its search space is highly discontinual and multimodal.
In this work a parallel computational system based on Ant Colony System (ACS) called Artificial-Ant-Colony Networks is introduced to solve the nuclear reactor core fuel reload optimization problem.
ACS is a system based on artificial agents that uses the reinforcement learning technique and was originally developed to solve the Traveling Salesman Problem, which is conceptually similar to the nuclear fuel reload problem.

ÍNDICE

CAPÍTULO 1

1. Introdução

O processo de recarga de um Reator Nuclear a Água Pressurizada (RNAP) ocorre toda vez que a queima dos elementos combustíveis (EC) no núcleo do reator atinge um determinado valor em que não é mais possível manter o reator crítico produzindo energia à potência nominal.

Os EC são descarregados do núcleo e armazenados em uma Piscina de Combustível Usado (PCU). Os EC com baixa concentração de $_{92}U^{235}$ são mantidos definitivamente na PCU, enquanto os EC com maiores concentrações de U^{235}, remanescentes do ciclo anterior e EC novos irão compor o núcleo do ciclo seguinte.

O problema de otimização da recarga consiste em determinar o posicionamento destes EC no núcleo do reator de forma otimizada, ou seja, minimizar a relação custo benefício dos EC, aproveitando-se ao máximo a queima dos EC e satisfazendo restrições de simetria e de segurança.

A otimização de recargas de RNAP é um problema do tipo NP-Completo, ou seja, a dificuldade deste problema cresce exponencialmente com o número de EC no núcleo do reator. O núcleo do reator de Angra 1, por exemplo, que contém 121 EC, gera um número de combinações de núcleo de aproximadamente 10^{273}. Todavia, como será visto neste capítulo, existem no núcleo simetrias de 1/4 e 1/8 e regras de colocação dos EC, este número cai para aproximadamente 10^{25}, que ainda assim é extremamente alto

para tentarmos resolver o problema por enumeração. Com esse número de combinações levaríamos aproximadamente 10^{19} anos para testarmos todas estas combinações, pois o tempo de execução de um código de física de reatores para avaliar uma possibilidade de solução é de aproximadamente 2 minutos. Além destas dificuldades, este problema tem características não lineares, descontinuidades, múltiplos máximos e mínimos locais no seu espaço de soluções (GALPERIN, 1995).

A solução deste problema é dependente dos objetivos a serem alcançados, que são basicamente dois: a minimização do fator de pico de potência radial do núcleo ou a maximização da duração do ciclo. Depois de feita a escolha do objetivo temos duas escolhas do esquema de carregamento, tipo *out-in* ou tipo baixa fuga, que serão expostos a seguir.

Existem diferentes estratégias para a recarga de combustível nuclear que podem determinar a distribuição de potência como função da queima para serem avaliadas por cálculos de queima. Geralmente, para se resolver este problema utiliza-se um esquema cíclico, em que somente uma parte dos EC é recarregada a cada ciclo (geralmente 1/3 por ciclo). Entre essas estratégias, as mais tradicionais são a estratégia tipo *out-in* e a tipo baixa fuga (MACHADO, 2001).

Se o objetivo for a minimização do fator de pico de potência radial, a estratégia a ser utilizada será do tipo *out-in*, os EC novos serão carregados na periferia do núcleo. Como estes estão fixos nas posições de periferia do núcleo, diminui-se o espaço de soluções, pois o número de EC para serem combinados é menor.

Já, se o objetivo for a maximização do comprimento do ciclo, procura-se inserir EC novos em posições mais internas do núcleo e EC mais queimados em posições mais externas. Neste tipo de carregamento, a fuga de nêutrons do núcleo é reduzida. Este esquema de carregamento, além de maximizar o comprimento do ciclo, diminui o desgaste do vaso do reator, cuja estrutura é danificada pelo fluxo de nêutrons rápidos, podendo ocasionar trincas no decorrer dos ciclos de operação. Este é um fator preponderante para a vida útil de uma usina nuclear (CHAPOT, 2000).

A recarga de RNAP deve obedecer às regras de simetria existentes nos reatores (CHAPOT, 2000). Há dois eixos de simetria principais, nas direções norte-sul e leste-oeste, e dois secundários, também chamados diagonais. Os dois eixos principais dividem o núcleo em quatro quadrantes, daí a simetria de ¼ de núcleo. Os dois eixos principais e as duas diagonais dividem o núcleo em oito partes simétricas, daí a simetria de $^{1}/_{8}$.

No caso do reator de Angra 1, Figura 1.1, os EC que compõem o núcleo podem ser classificados quanto à simetria em:

- Elemento Central.
- Quartetos – Conjuntos de 4 EC simétricos pertencentes a eixos de simetria.
- Octetos – Conjuntos de 8 EC simétricos, não localizados sobre os eixos de simetria.

Esta classificação simplifica o problema, pois quanto menor for o número de EC participantes da otimização, menor o espaço de busca das soluções do problema. Em outras palavras, menor a complexidade.

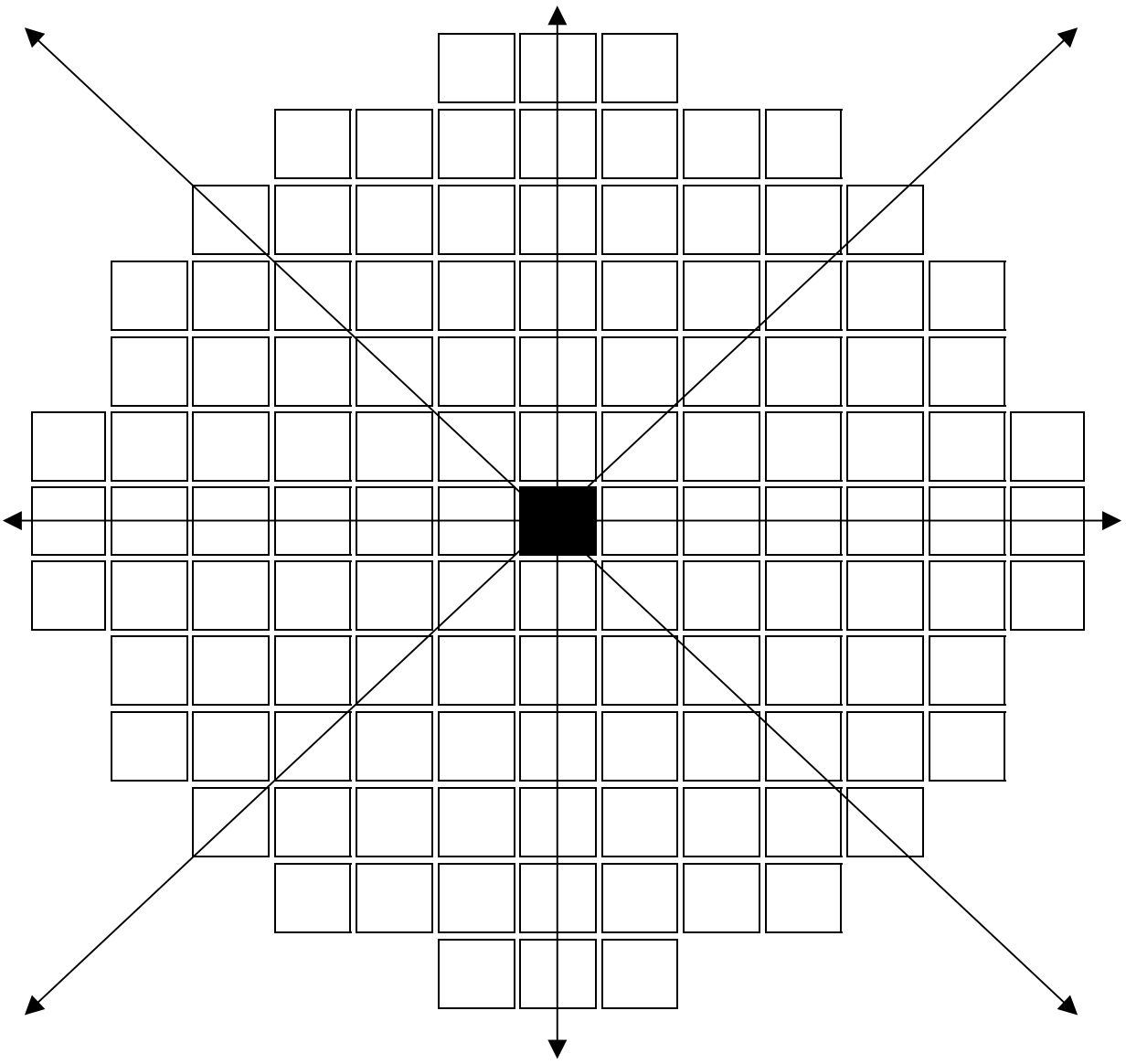

Figura 1.1 – Núcleo do reator de Angra 1

Durante décadas o problema de otimização da recarga do combustível nuclear foi solucionado somente através de otimização manual, onde especialistas utilizavam seus conhecimentos e experiência para construírem configurações de núcleo do reator, testando-as para verificar se as mesmas atendiam às restrições de segurança da usina.

A evolução verificada na informática a partir da década de 80, com o surgimento de processadores cada vez mais rápidos, propiciou o desenvolvimento de trabalhos que

procuravam substituir o especialista na otimização da recarga nuclear, buscando assim um processo automatizado.

Um trabalho de grande relevância, desenvolvido no início dos anos 90, que demonstrou a viabilidade de uma otimização automatizada, foi o código computacional FORMOSA – *Fuel Optimization for Reloads: Multiple Objectives by Silmulated Annealing* – (KROPACZEK E TURINSKY, 1991). Este código utiliza a técnica de *Simulated Annealing* (KIRKPATRIK et al, 1983) para realizar o processo de otimização na busca de soluções para o problema da recarga nuclear.

Uma nova abordagem para o código FORMOSA, proposta em 1992, substitui o método de busca *Simulated Annealing* (SA) pela técnica dos Algoritmos Genéticos (AG) (HOLLAND, 1975) no processo de otimização, surgindo o código FORMOGA (POON e PARKS, 1992).

Com o indicativo da superioridade dos Algoritmos Genéticos sobre o Simulated Annealing na solução do problema da recarga nuclear, surgiram outros códigos, que fizeram uso da técnica dos AG para realizar a otimização automatizada.

Em 1995, o código CIGARO – *Code Independent Genetic Algorithm Reactor Optimization System* – (DECHAINE e FELTUS, 1995) demostrou que o AG é uma ferramenta computacional eficiente para a solução do problema da recarga nuclear.

Os ciclos 7 e 9 da usina Angra 1, foram otimizados pelo sistema GENESIS/ALGER/ANC (CHAPOT, 2000).

Em 2001 foi desenvolvida uma tese de doutorado aplicando o algoritmo ANT-Q (MACHADO, 2001) em conjunto com o código de Física de Reatores RECNOD (CHAPOT, 2000) para otimização do ciclo 7 da usina Angra 1. Em função dos bons resultados, esta pesquisa apresenta uma nova abordagem dos algoritmos baseados em colônias de formigas aplicados ao problema da recarga de reatores nucleares.

No presente trabalho é apresentada uma nova modelagem computacional para realizar a recarga nuclear baseado em algoritmos paralelos de colônias de formigas, gerando o sistema denominado Redes Conectivas de Colônias Artificiais (RCCA) e seus resultados são comparados com os obtidos pelo sistema GENESIS/ALGER/RECNOD (CHAPOT, 2000), pois os resultados de MACHADO (2001) não foram reproduzidos.

Faremos a otimização da recarga do núcleo do reator utilizando implementações paralelas, modelo de ilhas (CANTÚ-PAZ, 1999) aplicado ao Sistema de Colônia de Formigas (SCF), com a migração das melhores soluções de cada ilha. A técnica de ilhas possibilita não apenas o fracionamento do esforço computacional, mas também tem conduzido a melhores resultados (CANTÚ-PAZ, 1998).

O SCF é um sistema baseado em Agentes Artificiais (RUSSEL e NORVIG, 1995) e que faz uso da técnica de Reforço de Aprendizagem (KAELBLING et al, 1996). Este algoritmo foi desenvolvido inicialmente para solucionar o Problema do Caixeiro Viajante (PCV) que é um problema combinatório conceitualmente similar ao da recarga nuclear.

Os excelentes resultados obtidos pelo SCF para o PCV (DORIGO E GAMBARDELLA, 1997), inclusive quando comparado com o AG, o tipo de programação utilizada - intrinsecamente paralela, de fácil compreensão e manutenção (agentes artificiais que executam apenas tarefas simples) - e a possibilidade do uso de uma heurística local para o problema motivaram sua aplicação ao problema da recarga nuclear (MACHADO, 2001).

No Capítulo 2, que é uma apresentação dos algoritmos baseados em colônias de formigas, mostra-se, inicialmente, uma introdução sobre processos naturais e Sistemas de Colônias de Formigas (SCF). Em seguida apresenta-se a motivação para se trabalhar com algoritmo baseados em formigas. Um tópico sobre otimização por Sistemas de Colônias de Formigas. Um tópico explicando o funcionamento das formigas artificiais. Em seguida é dada uma explicação de como funcionam algoritmos baseados em formigas.

No Capítulo 3, que é dedicado à apresentação de modelos paralelos, inicialmente apresenta-se a motivação para a implementação paralela. Logo após é apresentado o Modelo paralelo utilizado neste trabalho que é o Modelo de Ilhas. Em seguida se apresenta o funcionamento deste modelo. Logo após é apresentada uma descrição do funcionamento do SCF e uma adaptação deste modelo de paralelização ao SCF.

No Capítulo 4, que é dedicado à aplicação do sistema Redes Conectivas de Colônias Artificiais ao Problema do Caixeiro Viajante e seus resultados. O RCCA foi aplicado a três PCV, um de dificuldade mínima, um de dificuldade média e um de grande dificuldade. São apresentados os resultados em gráficos e tabelas.

O Capítulo 5 é o capítulo central dedicado à modelagem do RCCA ao problema da recarga. Na introdução é feita uma analogia entre o PCV e o problema da recarga. Em seguida são mostradas algumas diferenças entre estes dois problemas. Logo após são apresentadas as heurísticas utilizadas nos testes deste trabalho. Por último é dada uma explicação sucinta sobre o código de física de reatores utilizado.

No Capítulo 6 são apresentados os resultados obtidos pela aplicação do RCCA ao problema da recarga utilizando-se o código de física de reatores RECNOD.

Finalmente, no Capítulo 7, são apresentadas a conclusão e as propostas para trabalhos futuros.

CAPÍTULO 2

2. Algoritmos Baseados em Colônias de Formigas

2.1 Introdução

A pesquisa sobre modelos computacionais inteligentes vem, nos últimos anos, se caracterizado pela tendência em buscar a inspiração na natureza, onde existem inúmeros exemplos vivos de processos considerados inteligentes. Para cientistas de computação, matemáticos e engenheiros, muitas das soluções que a natureza encontrou, para problemas complexos de adaptação, fornecem modelos práticos interessantes.

Embora não se possa afirmar que tais soluções sejam todas ótimas, não há a menor dúvida que os processos naturais, em particular os relacionados diretamente com os seres vivos, sejam extremamente bem concebidos e adequados ao nosso mundo.

Físicos, biólogos e outros cientistas tentam desvendar os princípios que regem os fenômenos da natureza, enquanto que os matemáticos, cientistas de computação e engenheiros buscam idéias que possam ser copiadas ou pelo menos imitadas, e então aplicadas a problemas que a ciência atual ainda não consegue resolver satisfatoriamente, como é o caso da otimização da recarga de reatores nucleares.

Inspirado na natureza, a área de otimização teve um considerável progresso em estudos baseados em tais modelos. Um dos principais motivos que levam ao estudo

destes modelos é o fato de que a maioria dos problemas de otimização de larga escala não possuem uma solução ótima em um tempo viável, podendo apenas ser resolvido de forma aproximada. No contexto de otimização, grande parte dos problemas é dita NP-Completo (um problema não polinomial, indicando que o espaço de busca por soluções cresce exponencialmente com as dimensões do problema) (BURIOL, 2000).

A estratégia de colônia, inspirada no comportamento de colônias de formigas no processo de busca de alimento, ou seja, no comportamento coletivo de colônias de insetos opera com a idéia da comunicação indireta explorada pelas sociedades de insetos e formam algoritmos distribuídos de multi-agentes.

No início da década de noventa, desenvolveu-se um novo algoritmo para otimização combinatória. Este algoritmo foi inspirado na observação das colônias de formigas e, por isso, foi denominado *Ant System*. O *Ant System* foi aplicado, com sucesso, a problemas combinatoriais complexos como o Problema do Caixeiro Viajante (PCV) (DORIGO, GAMBARDELLA, 1997) e o Problema de Alocação Quadrática (PAQ) (GAMBARDELLA et al, 1999), e tem sido aplicado a vários problemas de otimização de cadeias de telecomunicações, roteamento de veículos, distribuição de tarefas e principalmente em problemas de otimização combinatória (BOECHEL, 2003).

2.2 Motivação

A motivação para trabalhar com a otimização de estruturas utilizando algoritmos baseados em formigas, originou-se do conhecimento de fatos curiosos e interessantes

que ocorrem com perfeição na natureza e que levaram os seres humanos a se inspirarem na busca de soluções para problemas do seu meio. Podemos citar vários exemplos destas inspirações decorrentes da natureza: pássaros levaram aos aviões, morcegos aos sonares, etc.

O fato curioso na natureza foi a descoberta realizada por etólogos que ao estudarem o comportamento de colônias de formigas: observaram que mesmo sendo elas seres tão simples e irracionais possuem mecanismos naturais de otimização. Isto é, são capazes de encontrar um caminho mais curto entre o formigueiro e uma fonte de alimento, sem usar sugestões visuais, mesmo que ocorram mudanças no ambiente original como a introdução de um obstáculo.

A explicação para a descoberta do caminho ótimo a ser percorrido pelas formigas é que elas depositam, durante suas caminhadas, uma certa quantidade de uma substância chamada feromônio, aproximadamente com taxa constante, e preferem seguir uma trajetória probabilisticamente mais rica em tal substância.

Quando o caminho original é interrompido pela introdução de um obstáculo, as mesmas se dividem pelas alternativas possíveis e, com o decorrer do tempo, como pelo menor caminho terão passado mais formigas, ficando o mesmo com uma maior concentração de feromônio, este novo trajeto será então adotado.

É bom destacar que o comportamento exibido pelas formigas em encontrar o menor caminho entre o formigueiro e a fonte de alimento é uma propriedade da

coletividade, ou seja, da colônia de formigas. Uma formiga, isoladamente, contribui para esse resultado, mas não é capaz de produzí-lo sozinha.

A idéia e a necessidade de otimização, exemplificada por simples formigas, em conjunto com a seleção natural dos mais aptos no processo de evolução, torna-se um campo interessante pelos inúmeros resultados de sucesso apresentados nas mais diferentes áreas de aplicação, uma alternativa de grande força por suas características: robustez, flexibilidade, eficácia e, principalmente, simplicidade.

A seguir, temos uma explicação com detalhes de como as formigas são capazes de encontrar o menor caminho entre o formigueiro e a fonte de alimento, com a ajuda do depósito de feromônio.

Inicialmente as formigas deixam o formigueiro de forma aleatória e isotrópica. A partir daí, as formigas que encontraram comida passam a depositar feromônio no solo para demarcar o caminho para que outras possam achá-lo também. Ocasionalmente, outras formigas, podem achar outros caminhos que levem-nas ao alimento, mas somente os caminhos mais curtos serão utilizados pela colônia, devido ao reforço do feromônio nos caminhos mais curtos ser maior do que nos caminhos mais longos, como veremos na Figura 2.1.

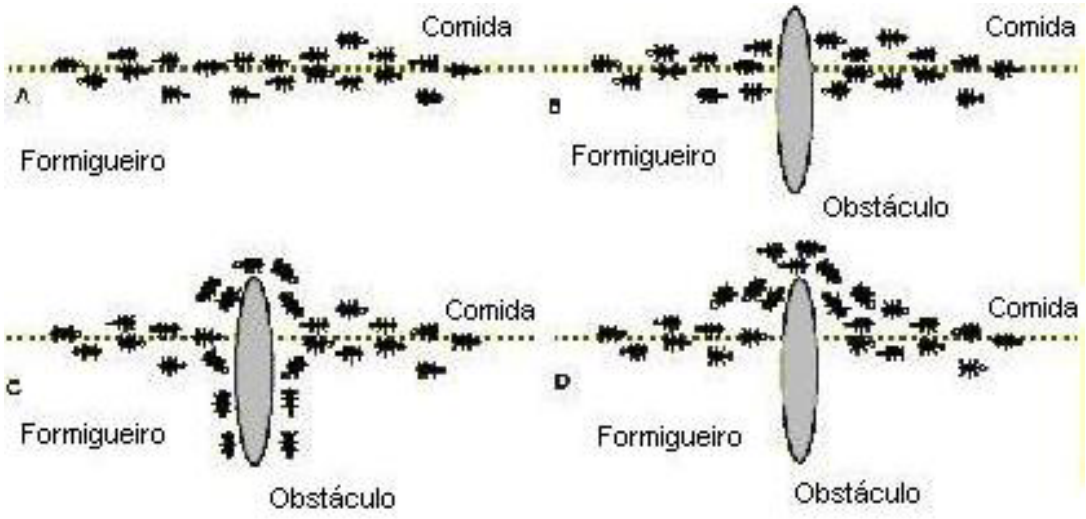

Figura 2.1 – Encontrando um novo melhor caminho

A explicação é bastante simples: os trechos mais curtos recebem mais feromônio do que os mais longos e como a quantidade de feromônio é proporcional à atração das formigas, locais com mais feromônio irão atrair mais formigas, com isso vai existir mais feromônio. Consequentemente, os trechos mais curtos são rapidamente achados.

Para a compreensão do mecanismo pelo qual as formigas sempre reforçam o caminho mais curto, será descrita uma situação fictícia e bastante simples onde existem apenas duas possíveis rotas para se contornar um obstáculo e se chegar à fonte de alimento, como mostra a Figura 2.2.(MACHADO, 2001).

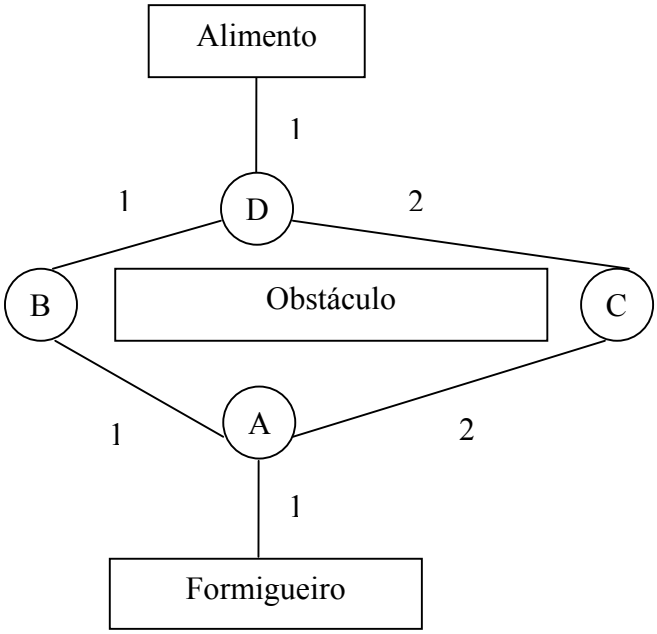

Figura 2.2 – Representação da rota formigueiro-alimento

Suponhamos que as formigas saem do formigueiro e encontrem a bifurcação A com um obstáculo. Neste ponto há dois caminhos a seguir: 1) o caminho ACD que tem 4 unidades de comprimento e 2) o caminho ABD que tem 2 unidades de comprimento.

Suponhamos também que todas as formigas movem-se com velocidades semelhantes e depositem quantidades de feromônio semelhantes, então, as formigas movem-se a 1 unidade de comprimento por 1 unidade de tempo e depositam uma unidade de feromônio em cada unidade de comprimento percorrido. Inicialmente não há nenhum rastro de feromônio.

Em t = 0, 20 formigas deixam o formigueiro e se movem 1 unidade de comprimento até chegarem à posição A. Elas escolherão ir para a esquerda ou para a direita com a mesma probabilidade. Assim, em média, 10 formigas irão para a esquerda e 10 para a direita.

Em t = 4, as primeiras 10 formigas que seguiram a rota ABD, encontrarão o alimento e retornarão.

Em t = 5, os dois grupos de 10 formigas se encontrarão em D. Nesta fase a quantidade de feromônio depositada em BD é igual à depositada em CD, assim novamente 10 formigas deverão selecionar, probabilisticamente, um dos dois trechos. Consequentemente, cinco formigas escolherão o trecho BD e as outras cinco escolherão o CD.

Em t = 8, as primeiras cinco formigas voltarão ao formigueiro e teremos: cinco formigas em cada um dos trechos AC, CD, BD.

Em t = 9, as primeiras cinco formigas estarão fazendo o caminho de volta ao alimento e terão percorrido o trecho do formigueiro até a posição A, ou seja, terão percorrido uma unidade de comprimento. Neste momento estarão enfrentando, mais uma vez, a escolha de seguir o trecho AB ou o trecho AC. Porém agora, o rastro de feromônio em AB será de 20 unidades, enquanto em AC será de 15 unidades. Então mais formigas escolherão, de modo probabilístico, ir para a esquerda (trecho AB) do que ir para a direita (trecho AC), e assim este caminho será reforçado.

Conforme a continuação do processo, a diferença da quantidade de feromônio entre as duas rotas ficará cada vez maior, até que quase todas as formigas passem a selecionar a rota mais curta.

Na realidade, as formigas não têm apenas duas opções de caminho e sim uma imensa quantidade deles. Dessa forma, o comportamento de uma colônia de formigas é muito mais complexo que o exemplo exposto acima. Porém, este comportamento, segue o mesmo princípio apresentado: Quanto mais formigas seguirem um rastro de feromônio, mais atraente este rastro se tornará para as outras formigas. Assim, a probabilidade de uma formiga seguir um determinado caminho é proporcional ao número de formigas que já o seguiram.

O processo de reforço, que emerge do comportamento coletivo das formigas, é o que permite a elas minimizarem a distância efetiva entre o formigueiro e a fonte de alimento. É esta característica que oferece inspiração para a aplicação dos princípios utilizados pelas formigas na solução de problemas de otimização, como por exemplo, o Problema do Caixeiro Viajante (PCV).

Essa inspiração levou ao desenvolvimento do *Ant System* (COLORMI et al, 1991) e, por consequência, de todos os outros algoritmos pertencentes aos sistemas de Otimização por Colônia de Formigas.

2.3 – Otimização por Colônias de Formigas

Insetos que vivem em sociedade, tais como formigas e abelhas, sobrevivem em praticamente toda a Terra, sendo que acompanharam a evolução da Terra. Sem dúvida, sua organização social, em particular o desenvolvimento genético do comprometimento

de cada indivíduo para a sobrevivência da colônia, é um fator chave que sustenta seu sucesso.

Além disso, estas sociedades de insetos exibem a fascinante propriedade que as atividades dos indivíduos, assim como as atividades da sociedade como um todo, não são reguladas por nenhuma forma de controle centralizado. Forças evolucionárias geraram indivíduos que combinaram um total comprometimento com a sociedade junto com comunicação específica e habilidades de ação que estimulam o surgimento de padrões complexos e comportamentos em relação ao nível global (DORIGO et al, 2002).

Dentre os insetos que vivem em sociedade, as formigas podem ser consideradas a família que teve mais sucesso. Existem cerca de nove mil espécies diferentes (DORIGO et al, 2002), cada uma com diferentes conjuntos de características especializadas que lhes permitem viver em grande número e, praticamente, em qualquer lugar.

A observação e estudo de formigas e sociedades de formigas há muito tempo vem atraindo a atenção de etólogos e leigos, mas nos últimos anos, o modelo de organização e interação das formigas tem chamado o interesse dos engenheiros e cientistas da computação (DORIGO et al, 2002).

A presença simultânea e única das características da sociedade das formigas, tais como: autonomia dos indivíduos, controle completamente distribuído, direção e comunicação mediada pelo meio, surgimento de comportamentos complexos com

relação ao repertório de uma única formiga, estratégias coletivas e cooperativas, e organização própria, têm feito dessa sociedade um atraente e inspirador modelo para construir novos algoritmos e novos sistemas multi-agentes.

Nos últimos anos, sociedades de formigas têm fornecido o impulso para um crescente corpo de trabalho científico, nos campos de robótica, pesquisa operacional, e telecomunicações (BOECHEL, 2003). Pesquisadores e cientistas de todo o mundo têm feito progressos significantes, tanto na parte teórica quanto em implementações, possibilitando assim dar a este campo de pesquisa uma base sólida. Além disso, tem-se mostrado que o "caminho das formigas", quando cuidadosamente construído e estudado pode resultar em aplicações de sucesso para muitos problemas do mundo real (DORIGO et al, 2002).

Assim como no campo dos algoritmos genéticos (HOLLAND, 1975) e redes neurais (HAYKIN, 1994) para citar dois exemplos, a natureza parece oferecer uma fonte de idéias para o projeto de novos sistemas e algoritmos.

A estratégia empregada por sociedades de formigas na captura de alimentos tem inspirado um novo paradigma computacional. O modelo inicialmente proposto por DENEUBOURG et al, 1983, sugere que há um determinado grau de aleatoriedade na comunicação das formigas que minimiza o tempo de captura do alimento distribuído entre várias fontes.

Comportamentos globais muito complicados podem resultar da interação de sistemas dinâmicos individuais muito simples. Esses comportamentos podem ser

observados em vários sistemas biológicos, como, por exemplo, em colônias de formigas. 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 eles possuem uma quantidade de "inteligência" individual suficiente para cooperarem de modo organizado.

O processo de procura e coleta de alimento realizado por colônias de formigas pode ser entendido como uma dessas atividades. O ambiente onde colônias de formigas vivem é mais ou menos previsível no tempo e no espaço. Pode-se comparar duas situações extremas em que fontes de alimento se apresentam: uma colônia de pulgões, como fonte durável (da ordem de meses), e um pássaro morto, que é uma fonte ocasional. A fim de explorar os pulgões, as formigas desenvolvem rotas estáveis entre o formigueiro e o ninho dos pulgões. Nessa situação, é benéfico que as formigas mantenham estruturas (rotas) permanentes com baixo nível de erro ou de flutuação, isto é, que poucas formigas "errem" o caminho (DENEUBOURG et al, 1983; DENEUBOURG et al, 1986).

Porém, depender de uma única fonte de alimento é muito arriscado para a sobrevivência da colônia. Assim, também é interessante explorar o pássaro morto, como fonte extra de alimento. Para isso, algumas formigas têm que "errar" o caminho original, isto é, deixar as rotas estáveis para que possam descobrir, ao acaso, a ave morta (DENEUBOURG et al, 1983; DENEUBOURG et al, 1986).

As formigas, como outros seres vivos, desenvolveram interações e mecanismos de comunicação que podem apresentar graus diversos de aleatoriedade que variam com

a espécie e com as condições do meio. Um desses mecanismos é baseado na deposição de feromônio no percurso realizado, geralmente entre o formigueiro e a fonte de alimento.

Feromônio é uma substância produzida pelas formigas cuja função é estabelecer um padrão de identificação e de comunicação entre elas. Uma quantidade inicial de feromônio depositado na trilha fará com que essa trilha seja detectada por outras formigas e, à medida que caminham pela trilha em direção à fonte de alimento, as formigas recém atraídas também depositam feromônio. Tem-se, a partir daí, um processo de realimentação positiva no qual, quanto mais feromônio estiver presente na trilha, mais formigas perdidas serão atraídas e maior será a quantidade de feromônio depositado na trilha, deixando o sistema mais determinístico (BOECHEL, 2003).

Nos últimos anos tem se notado o surgimento de um significativo conjunto de técnicas e algoritmos computacionalmente muito eficientes, denominados heurísticos ou aproximativos, baseados em técnicas de otimização de busca local para o tratamento de problemas NP-Completos. Embora eficientes, esses algoritmos não garantem uma solução ótima do problema devido à sua complexidade. Isso demonstra cada vez mais a importância de se encontrar algoritmos aproximados para resolver esses tipos de problemas (GOLDBARG et al, 2000).

O comportamento natural das formigas inspirou a construção de um algoritmo no qual um conjunto de formigas artificial coopera para a solução de um problema através da troca de informação via feromônio depositado sobre as trilhas. Esse algoritmo tem sido aplicado em problemas de otimização combinatória, como o

Problema do Caixeiro Viajante (PCV), que consiste em se visitar *n* cidades, sem repeti-las, e utilizando-se do menor caminho possível, como veremos em mais detalhes mais à frente (DORIGO, GAMBARELLA, 1997). A analogia adotada advém do fato de as formigas conseguirem restabelecer rotas interrompidas por obstáculos, de maneira que a nova rota é a mais curta possível.

Nesse algoritmo, formigas "virtuais" são enviadas para várias cidades e, após cada iteração, a quantidade de feromônio em cada trilha é atualizada proporcionalmente à distância percorrida pelas formigas, selecionando o percurso ótimo entre as cidades (DORIGO e GAMBARELLA, 1997).

Ao estudar insetos sociais como vespas, abelhas e em particular formigas, podemos ver que as colônias são muito organizadas. Entretanto parece que os insetos individualmente pouco sabem o que há ao redor deles, caminham uns atrás dos outros, fazendo todos as mesmas coisas. De alguma maneira na interação entre essas entidades muito simples, padrões emergem. O termo usado para este fenômeno é auto-organização (KAUFMAN, 1995).

Auto-organização é basicamente um conjunto de mecanismos de estruturas dinâmicas e aparecem em um nível mais alto, porém só aparecem devido às interações entre componentes em um outro nível mais baixo. Nestes meios não há nenhum supervisor para guiar os níveis mais baixos. Como resultado disto, as decisões feitas por estes componentes estão somente baseadas em informação local. As formigas não têm acesso ao conhecimento global contido no sistema como um todo.

Sistemas assim são achados em vários lugares e em diferentes níveis na natureza. Isto é uma indicação de que algoritmos baseados nestes sistemas podem ser muito poderosos. Podemos ver isso pela seleção natural, todos os sistemas são eficientes e robustos e apesar das dificuldades e mudanças no ambiente em que vivem, continuam vivendo e sempre se adaptando. Se eles não forem bastante eficientes e robustos, podem ser extintos e substituídos por outras espécies.

Como as formigas tiveram bastante êxito na evolução da Terra, é visto que esta auto-organização acontece em vários níveis. Por exemplo, dentro da colônia, umas estão trabalhando enquanto outras estão fora procurando comida, são formadas redes de rastros entre o ninho e várias fontes de comida. Barreiras naturais, normalmente, não representam nenhum problema porque as formigas acham o modo como desviar ao redor deles. Estas redes de rastros normalmente são muito eficientes. Isso é uma das razões principais que inspira o uso de colônias de formigas como um exemplo para técnicas de otimização.

De acordo com BONABEAU et al, 1999, há quatro aspectos importantes, mostrados com exemplos específicos em colônias de formigas:

Avaliação positiva - Reforçar as soluções melhores. É necessário o uso de algum modo de avaliar as experiências positivas. Para formigas que procuram comida, os rastros deixados pelas outras é uma forma de avaliação positiva. Os rastros de feromônio deixados são maiores para fontes de comida, e assim mais formigas irão lá, o que sucessivamente aumentará a quantia de feromônio daquela trilha.

Avaliação negativa - É preciso contrabalançar a avaliação positiva e estabilizar o padrão coletivo. Um exemplo é o esgotamento de fontes de comida. Essa avaliação negativa é embutida no algoritmo pela evaporação de feromônio, ou seja, os rastros diminuem com o passar do tempo.

Amplificação de flutuações - Algum fator aleatório é freqüentemente crucial para achar soluções melhores. Precisam ser descobertas novas fontes de comida e isso ocorre quando uma formiga às vezes tenta um caminho novo, ou se perde, em vez de seguir o caminho velho acaba descobrindo um nova fonte de comida.

Interações múltiplas - A auto-organização só funciona para o bem do conjunto se os indivíduos puderem interagir entre si. As formigas usam um rastro de feromônio como mecanismo para interagir umas com as outras. Os rastros são um exemplo de memória coletiva.

2.4 - Formigas Artificiais

As formigas artificiais são modelos modificados das formigas reais. Pois além de manterem algumas características das formigas reais, foram acrescentadas características que não possuem uma contrapartida na natureza. Este acréscimo de capacidade dado às formigas artificiais tem por objetivo fazer com que o sistema computacional seja mais eficiente na solução de problemas de otimização discreta.

Semelhante às formigas reais, os algoritmos da família Sistema de Colônias de Formigas (SCF) são compostos de uma população de formigas que cooperam globalmente para encontrar boas soluções para a tarefa que estão executando.

Cada uma das formigas constrói uma solução completa, porém, como acontece com as formigas reais, as boas soluções só serão obtidas com a cooperação mútua da colônia.

Enquanto as formigas reais depositam o feromônio para qualificar os caminhos no mundo real, as formigas artificiais modificam informações numéricas que representam o espaço de busca do problema tratado. Esta modificação nas informações numéricas é feita, por cada uma das formigas artificiais, levando em conta seus desempenhos atuais e pode ser acessada por qualquer outra formiga. Esta informação numérica é denominada de feromônio artificial e é através dela que uma formiga artificial se comunica com as outras.

As formigas artificiais também fazem uso de um importante mecanismo utilizado pelas formigas reais na busca do menor caminho entre o formigueiro e a fonte de alimento: a evaporação do feromônio. Esse mecanismo permite que a colônia de formigas esqueça lentamente seu passado, permitindo redirecionar a busca para novas direções. As formigas artificiais, ao fazerem uso deste mecanismo, se previnem para que não fiquem retidas em pontos de mínimos locais do espaço de busca.

As formigas artificiais, assim como as reais, aplicam uma política de decisão probabilística de movimentação para os próximos estados. Essa política utiliza somente

conhecimentos locais e é função de dois tipos de informação. A primeira parte é uma informação específica sobre o problema tratado, a heurística do problema (em uma analogia com as formigas reais, seriam informações sobre o terreno em que elas se movimentam para encontrar a comida) e a segunda parte é a informação introduzida pelas formigas no ambiente, ou seja, o rastro de feromônio.

As formigas artificiais fazem parte de um mundo discreto onde os seus movimentos consistem na transição de um estado discreto para outro estado discreto e ainda possuem uma memória de suas ações passadas.

2.5 Algoritmos Baseados em Colônias de Formigas

Conforme mencionado anteriormente, os algoritmos baseados em Colônia de Formigas são sistemas computacionais que utilizam formigas artificiais que simulam o comportamento das formigas reais.

Uma formiga artificial realiza operações simples com as capacidades definidas abaixo:

- Mantém uma memória dos lugares já visitados (capacidade de memória),
- Move-se aplicando uma Regra de Transição de Estado,
- Depois de cada movimento atualiza o espaço de busca localmente aplicando uma Regra de Atualização local,
- Depois de gerar uma solução completa, atualiza globalmente o espaço de busca aplicando uma Regra de Atualização Global.

Existem vários algoritmos baseados em Colônia de Formigas (STUTZLE, DORIGO, 2000), mas neste trabalho foi utilizado o *Ant Colony System* (ACS) ou Sistema de Colônia de Formigas (SCF) (GAMBARDELLA, DORIGO, 1996).

2.6 Sistema de Colônias de Formigas (SCF)

O SCF foi desenvolvido para aplicação em problemas de otimização combinatória do tipo NP-Completo (NP – Não polinomial), ou seja, a dificuldade deste problema cresce exponencialmente com o número de elementos, como é o caso do Problema do Caixeiro Viajante (PCV). A forma mais fácil de explicar o SCF é através de sua aplicação a este problema.

O PCV é um dos mais conhecidos na área de otimização. Trata-se de um caixeiro viajante que deve visitar um conjunto de cidades, partindo de uma cidade inicial, passando por todas as demais, uma única vez, e retornar a cidade de origem cobrindo a menor distância possível.

Na prática, um problema com *n* cidades tem seu espaço de busca definido pela Equação (2.1):

$$S = \frac{(n-1)\,!}{2} \tag{2.1}$$

onde S é o número de possíveis rotas que podem ser percorridas, o que dá um número formidável mesmo para poucas cidades. Para $n = 30$, por exemplo, há um total de $4{,}42 \times 10^{30}$ rotas distintas. Com um computador capaz de analisar um milhão de

rotas por segundo, a busca completa levaria o equivalente a 11 milhões de vezes a idade do universo, considerando-se que o universo possui aproximadamente 12 bilhões de anos.

A popularidade do problema do caixeiro viajante provavelmente se dá ao fato de que é um problema muito fácil de ser entendido e visualizado, mas porém muito difícil de ser resolvido. Muitos outros problemas da classe NP não são somente de difícil resolução, mas como também de difícil compreensão.

Para exemplos pequenos não se torna nenhum problema gerar todas as possíveis soluções e escolher o caminho mais curto. Mas quando o número de cidades aumenta, as possíveis soluções "explodem". É ai então que precisamos de métodos mais inteligentes e viáveis computacionalmente para resolver, ou pelo menos achar soluções aceitáveis.

A procura por soluções aceitáveis ou razoáveis é um outro modo de ver os problemas de otimização como o do PCV. Em vez de tentar achar a melhor solução realizando infinitos cálculos em uma quantidade de tempo que talvez nem se possa estimar, achar uma solução aceitável em um espaço de tempo mais curto do que seria necessário para resolver de fato o problema.

A estratégia de resolução usada nestes casos é uma estimativa dos custos envolvidos achando entre as possíveis soluções um substituto ótimo como resultado. O PCV é um dos mais importantes e estudados problemas de otimização. A importância de encontrar um algoritmo aproximado para este problema reside no fato de muitos

problemas que ocorrem na ciência da computação, e em muitas outras áreas, poderem ser modelados a partir do mesmo, como é o caso da recarga nuclear. Outra importante característica é a grande dificuldade de se achar uma solução exata para instâncias maiores, visto que é um dos problemas de otimização mais difíceis, dentre os conhecidos. Nas últimas décadas, vem crescendo o desenvolvimento de algoritmos aproximados, geralmente utilizando métodos heurísticos que forneçam soluções aceitáveis em tempo de processamento compatível com as necessidades.

O PCV pode ser definido matematicamente da seguinte forma:

Existem 2 tipos de PCV, os simétricos e os assimétricos. Nos PCV simétricos tem-se que à distância de ir da cidade *r* para a cidade *s* é igual à distância da volta, ou seja, à distância da cidade *s* para a cidade *r*. Já nos PCV assimétricos essas distâncias são diferentes. Matematicamente temos as Equações 2.2 e 2.3:

PCV simétrico,

$$\delta(r,s) = \delta(s,r) \tag{2.2}$$

PCV assimétrico,

$$\delta(r,s) \neq \delta(s,r) \tag{2.3}$$

Quando se aplica o SCF ao PCV (GAMBARDELLA, DORIGO, 1996), o agente *k* é uma formiga artificial que se move de cidade em cidade, até retornar à cidade de origem, construindo uma solução. A esse agente *k* está associada uma lista de cidades a visitar $J_k(r)$, onde *r* é a cidade atual do agente *k*. Neste trabalho, a palavra agente está

diretamente relacionada com formiga. Uma explicação mais detalhada de agente é mostrada no apêndice A.

O agente *k* escolhe uma cidade para deslocar-se através de uma regra de transição de estado descrita pela Equação (2.4) (GAMBARDELLA, DORIGO, 1996) a seguir:

$$s = \begin{cases} \max \{[FE(r,s)]^{\delta} \; x \, [HE(r,s)]^{\beta}\} & se \;\; q \le q_0 \\ Roleta & se \;\; q > q_0 \end{cases} \qquad (2.4)$$

onde:

- FE(r,s) é um valor real e positivo associado ao arco (r,s). FE é uma matriz quadrada cuja ordem é definida pelo número de cidades. Os valores FE(r,s) dessa matriz são modificados em tempo de execução do algoritmo e indicam quanto é vantajoso o movimento para a cidade *s* quando se está na cidade *r*. Esses valores são o rastro de feromônio artificial depositado pelas formigas na construção de seus caminhos.
- HE(r,s) é o valor da função heurística relativa ao movimento da cidade *r* para a cidade *s*. Esta função expressa uma informação puramente local, independente da viagem como um todo. Ela estará indicando qual o custo de se mover de uma cidade para outra, sem levar em consideração qual o caminho percorrido até o momento. Para o PCV, o valor de HE(r,s) é o inverso da distância entre a cidade *r* e a cidade *s*.
- Os parâmetros δ e β pesam a importância relativa entre o aprendizado dos agentes, com os valores FE, e os valores heurísticos.

- q é um valor escolhido aleatoriamente com probabilidade uniforme dentro do intervalo [0,1] e q_0 ($0 \leq q_0 \leq 1$) é um parâmetro de entrada do algoritmo. Quando $q \leq q_0$ a regra de decisão utiliza o conhecimento disponível sobre o sistema, ou seja, o conhecimento heurístico e o conhecimento adquirido através do rastro de feromônio artificial. Quando $q > q_0$ a regra de decisão passa a utilizar uma distribuição de probabilidades e com isso, passa a explorar mais o espaço de busca do problema. O parâmetro q_0 permite regular se a escolha do próximo estado será concentrada nas melhores soluções ou se irá explorar mais o espaço de busca.
- Roleta é uma variável aleatória selecionada de acordo com uma determinada distribuição de probabilidades. Algumas dessas distribuições serão discutidas na próxima seção.

A regra, representada pela Equação (2.4), define a política de decisão probabilística de movimento para os próximos estados, que é uma função local e leva em consideração duas informações básicas. A primeira delas, que são os valores FE, é o modo como os agentes alteram seu espaço de busca para beneficiar a descoberta dos menores caminhos. Essa informação é o feromônio artificial que está associado à técnica do reforço da aprendizagem. A segunda informação usada na escolha para o próximo movimento é a informação referente ao problema específico que está sendo otimizado, ou seja, a heurística.

Inicialmente, como ainda não há uma grande interação entre os agentes, os valores heurísticos predominam sobre os valores FE. Neste ponto, fazer uso de uma

heurística representativa do problema em questão é extremamente importante, pois o passo inicial do algoritmo será dado a partir dessa informação e não de modo aleatório como acontece em vários métodos de otimização, como, por exemplo, os Algoritmos Genéticos, cuja explicação mais detalhada se encontra no anexo B.

Conforme os agentes vão interagindo, eles passam a aprender valores FE que favorecem a busca da solução ótima e, então, esses valores FE passam a ter maior peso para a escolha do próximo movimento.

Os valores FE (a quantidade de feromônio) são modificados através da cooperação entre os agentes para favorecer a descoberta de boas soluções. As duas regras abaixo são responsáveis pela atualização desses valores.

Regra de atualização local

$$FE(r, s) = (1 - \rho) * FE(r, s) + \rho * FEzero \quad (2.5)$$

onde:

ρ é o parâmetro de evaporação de feromônio
FEzero é o valor inicial da quantidade de feromônio

A regra de atualização local, dada pela Equação (2.5), é utilizada após todos os agentes terem aplicado a regra de transição de estado e escolhido a próxima cidade a ser visitada. Dessa forma, essa atualização é aplicada enquanto a solução está sendo construída.

O objetivo da regra de atualização local é estimular a busca por novas regiões do espaço de busca. Ela atua no sentido de diminuir o valor de FE (r,s) quando o agente que estava na cidade *r* e escolheu a cidade *s* como sua próxima cidade, ou seja, os

valores de FE relativos aos arcos do caminho já escolhidos pelos agentes são penalizados para evitar uma convergência prematura.

A quantidade de feromônio dos arcos é diminuída lentamente para permitir que as formigas artificiais ampliem seus espaços de busca. Este processo é denominado evaporação do feromônio.

O segundo tipo de atualização é a regra de atualização global. Ela é aplicada após todos os agentes terem construído um caminho completo e este caminho ter sido avaliado por uma função objetivo. Para o PCV, a função objetivo é a distância total percorrida em cada viagem completa. Esta regra é considerada o reforço de aprendizagem do algoritmo. Em geral, o reforço pode ser de dois tipos, o reforço local e o reforço global.

O reforço local é dado pela Equação (2.6):

$$\text{Reforço local (r,s)} = \begin{cases} \dfrac{W}{L_k} & se \;\; (r,s) \in a \;\; rota \;\; do \;\; agente \;\; k \\ 0 & caso \;\; contrário \end{cases} \qquad (2.6)$$

Onde:

L_k é à distância percorrida pelo agente que realizou o menor caminho na iteração atual.

k é o agente que realizou a menor viagem.

W é um parâmetro, definido pelo usuário, que juntamente com o parâmetro α expressam a velocidade de aprendizado do algoritmo.

O reforço global é dado pela Equação (2.7):

$$FE(r, s) = (1 - \alpha) * FE(r, s) + \alpha * (W / \text{melhor resultado}) \quad (2.7)$$

onde:

α é um parâmetro de evaporação de feromônio

$$\text{Reforço global (r,s)} = \begin{cases} \dfrac{W}{L_{kg}} & se \quad (r,s) \in a \quad rota \quad do \quad agente \quad k \\ 0 & caso \quad contrário \end{cases} \quad (2.8)$$

Onde:

L_{kg} é a distância percorrida pelo agente que realizou o menor caminho em todas as iterações realizadas até o momento.

K_g é o agente que realizou a menor viagem de todas as iterações.

W é um parâmetro, definido pelo usuário, que juntamente com o parâmetro α expressam a velocidade de aprendizado do algoritmo.

Nos testes realizados neste trabalho e, consequentemente, na construção do algoritmo apresentado, foi utilizado como reforço de aprendizado o reforço local, pois os testes realizados por Gambardella e Dorigo (GAMBARDELLA, DORIGO, 1996) demostraram que com esse tipo de reforço os resultados obtidos são melhores que aqueles obtidos com o reforço global.

A seguir é descrito o Sistema de Colônia de Formigas (SCF) de forma simplificada.

O primeiro passo a ser dado é a fase de inicialização. Nesta fase, os valores FE assumirão um valor constante FEzero, cada um dos agentes será posicionado sobre uma cidade de acordo com um método de inicialização (geralmente inicialização aleatória) e a lista $J_k(r)$, que contém as cidades ainda não visitadas, também será inicializada.

O passo seguinte é cíclico. Cada agente escolherá uma cidade para se deslocar e, após essa escolha, haverá a atualização local dos valores FE. Este ciclo será repetido até que todas as formigas tenham completado sua viagem.

No terceiro passo, a distância percorrida *L* do caminho de cada um dos agentes será avaliada pela função objetivo, o reforço de aprendizagem será calculado e a atualização global dos valores FE será realizada.

Distribuição de probabilidades usadas nas regras de transição de estado.

O algoritmo SCF foi testado com duas distribuições de probabilidades que são as seguintes:

- Distribuição Pseudo-Randômica.

Na distribuição Pseudo-Randômica a variável Roleta da Equação (2.4) é uma cidade pertencente à lista de cidades ainda não visitadas, $J_k(r)$, escolhida aleatoriamente de acordo com uma distribuição uniforme. Após os testes desta distribuição, foi

constatado que esta distribuição fornecia resultados muito inferiores à distribuição Pseudo-Randômica-Proporcional, e esta distribuição foi descartada.

- Distribuição Pseudo-Randômica-Proporcional

Na distribuição Pseudo-Randômica-Proporcional a variável Roleta na Equação (2.4) é aleatoriamente selecionada de acordo com a Equação (2.9) (GAMBARDELLA, DORIGO, 1996)abaixo:

$$\text{Roleta} = \begin{cases} \dfrac{[FE(r,s)]^{\delta} \; x [HE(r,s)]^{\beta}}{\sum_{z \in J_k(r)} [FE(r,z)]^{\delta} \; x [HE(r,z)]^{\beta}} & se \quad s \in J_k(r) \\ 0 & se \quad s \notin J_k(r) \end{cases} \tag{2.9}$$

onde:

- FE(r,s) são os valores FE que representam o rastro de feromônio artificial.
- HE(r,s) é o valor da função heurística associada ao arco (r,s).
- Os parâmetros δ e β pesam a importância relativa entre o aprendizado, com os valores FE, e os valores heurísticos.
- $J_k(r)$ é uma lista que contém as cidades ainda não visitadas pelo agente k, quando ele está localizado na cidade *r*.

Esta distribuição expressa a probabilidade de um agente, estando na cidade *r*, escolher a cidade *s* como seu próximo movimento. É uma roleta aleatória similar à roleta utilizada nos Algoritmos Genéticos (Holland, 1975) para selecionar os indivíduos para a próxima geração.

Abaixo vemos o pseudo código do algoritmo SCF e do RCCA.

```
Inicialização dos parâmetros
Para cada iteração faça
        Cada formiga é posicionada em um nó
        Para cada passo faça
                Cada formiga aplica uma regra de transição de estado para
                incrementar a construção de uma solução e aplica a atualização
                local para modificar o espaço de busca localmente
        Até que todas as formigas tenham construído uma solução
        Aplica a regra de atualização global para atualizar o espaço de busca
        globalmente
Até condição de término
```

Figura 2.3 – Pseudo-código do Algoritmo SCF

```
Inicialização dos parâmetros
Se migrar = Verdade então fazer migração
Para cada ilha faça
        Para cada iteração faça
                Cada formiga é posicionada em um nó
                Para cada passo faça
                        Cada formiga aplica uma regra de transição de estado para
                        incrementar a construção de uma solução e aplica a atualização
                        local para modificar o espaço de busca localmente
                Até que todas as formigas tenham construído uma solução
                Aplica a regra de atualização global para atualizar o espaço de busca
                globalmente
        Até condição de término
Fim ilhas
```

Figura 2.4 – Pseudo-código do Algoritmo RCCA

CAPÍTULO 3

3 – Paralelização de Algoritmos Evolucionários

Neste capítulo, é apresentada a justificativa para a utilização de SCF paralelos. Apresentamos também, como objeto de estudo neste trabalho, uma técnica de paralelização chamada Modelo de Ilhas, que consiste em processar vários algoritmos em separado (um em cada ilha) e fazer migrações periódicas de indivíduos de uma ilha para outra, tentando assim manter uma diversidade populacional.

3.1 – Motivação para Implementação Paralela

Os Algoritmos Evolucionários são métodos eficientes de busca baseados no princípio de sobrevivência dos mais aptos (DARWIN, 1859). Esses métodos têm sido aplicados na solução de problemas em finanças, em engenharia e nas ciências em geral. Os Algoritmos Evolucionários são geralmente capazes de achar soluções satisfatórias em um tempo razoável, mas se forem aplicados a problemas de grande complexidade e porte, como é o caso da recarga de reatores nucleares, pode existir um consumo excessivo de tempo computacional para se achar uma solução aceitável, pois os códigos de física de reatores, por serem códigos iterativos, são lentos, para este tipo de aplicação. Consequentemente, têm havido vários esforços para se obter Algoritmos Evolucionários mais rápidos, e uma das opções mais promissoras é o uso de Algoritmos Evolucionários com implementações paralelas (CANTÚ-PAZ, 1998).

3.2 – Implementação Paralela do Modelo de Ilhas

A idéia básica da maioria dos algoritmos paralelos é dividir a tarefa em partes e resolvê-la simultaneamente utilizando vários processadores. Esse procedimento pode ser empregado nos Algoritmos Evolucionários de várias maneiras. Alguns métodos de paralelização usam uma população simples, enquanto outros dividem a população em várias subpopulações isoladas (CANTÚ-PAZ, E., 1998).

O Modelo de Ilhas aplicado a um Algoritmo Evolucionário paralelo com população múltipla, consiste em várias subpopulações, as quais trocam indivíduos de tempos em tempos, sendo esta troca de indivíduos, com uma determinada estratégia (DE LIMA, 2000), denominada migração, ver Figura 3.1.

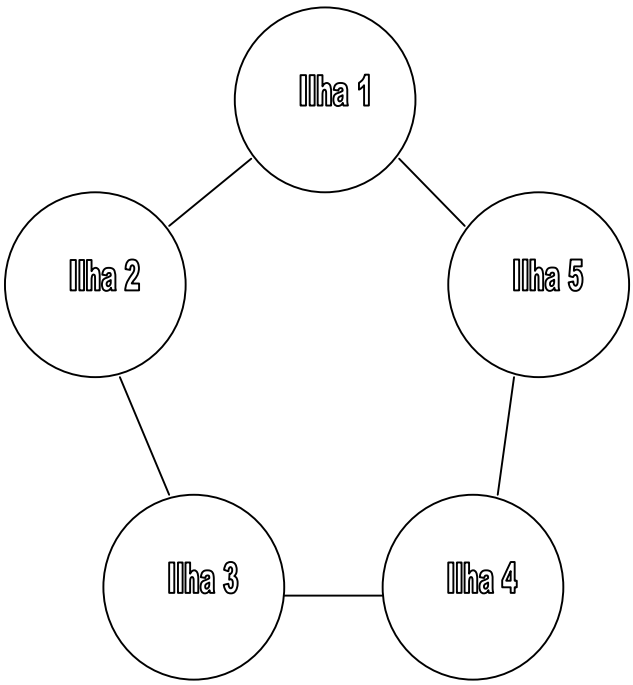

Figura 3.1 – Exemplo de Algoritmo Evolucionário paralelo com multipopulação de 5 ilhas dispostas em forma de anel.

O Modelo de Ilhas é um modo eficiente de implementarmos um Algoritmo Evolucionário de forma paralela, utilizando-se várias máquinas/processadores. Em uma implementação paralela do Modelo de Ilhas, cada máquina executa um Algoritmo Evolucionário e mantém sua própria população. As máquinas trabalham em conjunto e periodicamente trocam indivíduos de sua população em um processo chamado ***migração***.

Com a implementação do Modelo de Ilhas, podemos acelerar o processo de evolução do algoritmo, pois com o crescimento do número de máquinas serão obtidas respostas em tempo cada vez menores, até um limite em que o tempo de comunicação entre as máquinas atinja valores incompatíveis com a tarefa executada. Outra vantagem de se utilizar o Modelo de Ilhas é que com a migração (troca de informação, entre as ilhas), aumenta-se a diversidade populacional e com isso faz-se uma maior varredura do espaço de soluções.

3.3 – Funcionamento do Modelo de Ilhas

Uma população total de N indivíduos de um Algoritmo Evolucionário serial pode ser distribuída em I ilhas e cada ilha ser um Algoritmo Evolucionário independente, comunicando-se com as outras ilhas somente no momento da migração. Podemos ainda ter M máquinas e dividir tarefas entre estas máquinas. Com o processamento paralelo, diminuímos o tempo necessário para execução de uma tarefa. O Modelo de Ilhas introduz dois parâmetros novos: o Intervalo de Migração, que é o

número de gerações entre as migrações e o Tamanho da Migração, que é o número de indivíduos que migram entre as ilhas.

Modelos de Algoritmos Evolucionários paralelos têm obtido resultados superiores em relação aos modelos seriais, tanto em termos da qualidade da solução encontrada quanto em termos do número total de avaliações (DE LIMA, 2000). Uma das explicações para esta superioridade é que as ilhas mantêm um grau de independência e com isso exploram diferentes regiões do espaço de busca, trocando ainda informações através das migrações. Ou seja, a diversidade populacional é mantida.

A Figura 3.1 mostra um dos tipos de Modelo de Ilhas, neste caso um problema com 5 ilhas em forma de anel.

Para executarmos um Modelo de Ilhas seguimos os seguintes passos:

1. Definimos o intervalo de migração e o tamanho da migração.
2. Começamos a executar os Algoritmos Evolucionários simultaneamente em todas as ilhas. Essas ilhas são independentes e possuem todas o mesmo número de indivíduos.
3. Quando o número de gerações especificado (o intervalo de migração) tiver sido atingido fazemos a migração.
4. As migrações ocorrem entre todas as ilhas simultaneamente.
5. Os ***x***% melhores indivíduos da ilha doadora são copiados para a ilha receptora e os ***x*** % piores indivíduos da ilha receptora são removidos.

6. Este processo é repetido até a convergência do algoritmo.

É importante ressaltar que o modelo é sensível aos parâmetros, intervalo de migração e tamanho da migração. Se, por exemplo, o intervalo de migração ficar muito pequeno, o tempo de execução vai crescer por causa do aumento do número de comunicações entre as máquinas. Da mesma forma, se o tamanho da migração aumentar muito, existirão mais indivíduos para serem migrados entre as máquinas, gastando assim mais tempo de comunicação. Por outro lado, se o intervalo de migração ficar muito grande e o tamanho da migração muito pequeno, ganharemos em tempo de comunicação mas vamos começar a perder informações importantes na troca de indivíduos; o ideal é achar uma correlação ótima entre intervalo de migração e tamanho da migração, que dê a melhor solução, minimizando o tempo gasto pelo algoritmo.

3.4 – Modelo de Ilhas Aplicado ao SCF

O Modelo de Ilhas foi desenvolvido originalmente para os Algoritmos Genéticos (CANTÚ-PAZ, 1999) e neste trabalho este modelo foi adaptado ao SCF. O nome mais apropriado do Modelo de Ilhas Aplicado ao SCF é **Redes Conectivas de Colônias Artificiais (RCCA)**, pois este modelo é formado de vários formigueiros artificiais conectados à procura de alimento.

3.4.1 – Descrição Física do SCF

O SCF é um algoritmo que utiliza duas matrizes quadradas, cuja ordem é dada pelo número de cidades, no caso do PCV, ou pelo número de Elementos combustíveis, no caso da recarga nuclear. Uma dessas matrizes é a matriz Heurística, Figura 3.2, que representa o conhecimento prévio acerca do problema e que em nenhum momento é alterada, servindo apenas para fornecer o custo de se deslocar do elemento *r* para o elemento *s*.

Heurística	**1**	**2**	**3**	**4**	**5**
1	0	5	6	1	2
2	4	0	7	8	1
3	9	3	0	2	5
4	5	6	1	0	3
5	9	8	1	4	0

Figura 3.2 – Matriz Heurística 5x5

Essa matriz funciona de uma forma bastante simples, informando qual o custo de deslocar-se da cidade *r* (linhas) para uma cidade *s* (colunas). Os valores da diagonal são sempre nulos, pois significam deslocar-se de *r* para *r*.

A outra matriz é a matriz Feromônio, representada na Figura 3.3, que é a matriz que vai armazenar o aprendizado das formigas ao longo do processo de evolução do algoritmo e que é inicializada com todos os valores idênticos, pois as formigas ainda não iniciaram o aprendizado, e com o passar das gerações vai sendo modificada através

da técnica do reforço de aprendizado representado pelas Equações (2.5) e (2.7), que atualizam o feromônio de 2 formas.

A primeira forma, através da atualização local, dada pela Equação (2.5), consiste em não permitir que todas as formigas desloquem-se pelos mesmos caminhos. Quando a formiga que está na cidade *r* escolhe como destino a cidade *s*, é dado um decréscimo de feromônio neste arco (r,s), tornando este arco menos atraente às formigas subsequentes. Isto é uma forma de evitar que todas as formigas convirjam para os mesmos caminhos muito rapidamente, ou seja, uma forma de evitar uma convergência prematura.

A segunda forma é reforçar os melhores caminhos, como um todo, tornando-os mais atraentes. Esse reforço dá-se através de um acréscimo dos valores correspondentes a esse caminho na matriz Feromônio. Isto pode parecer uma situação antagônica à citada acima, mas devemos lembrar que o decréscimo se dá apenas nos arcos (r,s) com a primeira (Equação (2.5)) e o acréscimo se dá no caminho como um todo com a segunda (Equação (2.7)).

Feromônio	**1**	**2**	**3**	**4**	**5**
1	0	3	1	4	0
2	6	0	4	5	1
3	1	3	0	4	7
4	2	9	1	0	0
5	5	3	6	0	0

Figura 3.3 - Matriz Feromônio 5x5

A matriz feromônio representa o aprendizado que as formigas obtiveram após um dado número de gerações de evolução do algoritmo e representa um outro custo de deslocamento da cidade *r* para a cidade *s*. O custo representado não é mais o custo que conhecíamos *a priori* na matriz heurística, mas sim o custo que as formigas determinaram através de sua jornada.

A decisão acerca de para qual cidade deslocar-se leva em consideração estes dois custos em conjunto representados pela Equação (2.4). E com o passar das gerações o custo determinado pelas formigas começa a pesar mais na decisão e, com isso, a cooperação das formigas (agentes) torna-se a forma predominante de decisão do algoritmo.

Na Figura 3.4 vemos um exemplo de como a matriz Feromônio transforma-se em um caminho, de forma bastante simples.

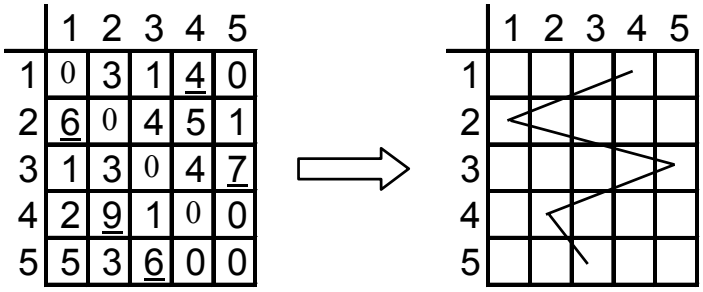

Figura 3.4 – Matriz Feromônio 5x5 e sua representação gráfica

Na Figura 3.4 a matriz feromônio representa o caminho 4-1-5-2-3-4, ou seja, em um PCV com 5 cidades o caminho representado seria sair da cidade 4, pois na linha 1 o maior valor de feromônio é na coluna 4, ir para 1, pois na linha 2 o maior valor de feromônio é a coluna 1, seguir para 5, pois na linha 3 o maior valor é a coluna 5, ir para

2, pois na linha 4 o maior valor é a coluna 2 e para 3, pois na linha 5 o maior valor é a coluna 3, voltando finalmente à cidade inicial.

3.4.2 – Adaptação do Modelo de Ilhas ao SCF

O Modelo de Ilhas, como já dito, foi elaborado originalmente para os Algoritmos Genéticos. A diferença básica do Modelo de Ilhas aplicado ao SCF e o aplicado aos Algoritmos Genéticos (apêndice B) é que nos AG a migração é feita com uma quantidade de indivíduos da população dependente da taxa de migração que vai participar da próxima população aplicando-se sobre eles os operadores genéticos ou não, e no SCF esta migração é feita como se segue.

No SCF existem outros tipos de troca de informação (MIDDENDORF et al, 2000), mas a apresentada neste trabalho foi a seguinte:

- Executar os algoritmos independentes até o número de gerações estipulado pelo intervalo de migração.
- Enviar a formiga que obteve o melhor resultado, naquela geração, da ilha doadora para a ilha receptora e fazer a atualização da matriz de Feromônio da ilha receptora, obrigatoriamente, com esta melhor formiga.
- Após esta troca de informações os algoritmos caminham normalmente e isoladamente até a próxima troca.

A escolha das ilhas doadoras e receptoras obedecem a uma ordem dinâmica, mostrada na Figura 3.5.

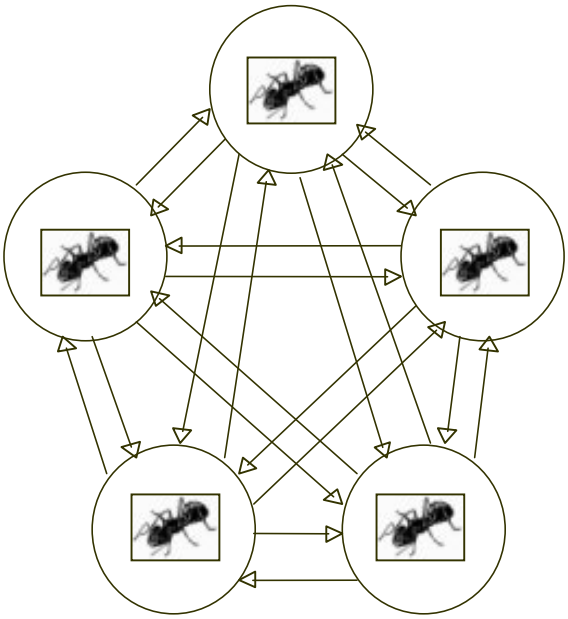

Figura 3.5 – Ordem de migração com 5 ilhas

No exemplo ilustrado acima, na primeira parada para migração, as migrações seriam feitas da ilha 1 para a ilha 2, da 2 para a 3, da 3 para a 4, da 4 para a 5 e da 5 para a 1. Na segunda parada para migração a ordem seria da ilha 1 para a ilha 3, da 3 para a 5, da 5 para a 2 da 2 para a 4 e da 4 para a 1. Na terceira parada a ordem seria da ilha 1 para a ilha 4, da 4 para 2, da 2 para a 5, da 5 para a 3 e da 3 para a 1. Na quarta parada a ordem seria da ilha 1 para a ilha 5, da 5 para a 4, da 4 para a 3, da 3 para a 2 e da 2 para a 1. A migração dinâmica é uma tentativa de espalhar isotropicamente as boas soluções por todas as ilhas.

O intervalo de migração é um parâmetro de grande importância no processo de busca. A definição desse parâmetro é crítica e seus valores ótimos são encontrados empiricamente.

CAPÍTULO 4

4. Resultados obtidos pela aplicação das Redes Conectivas de Colônias Artificiais (RCCA) ao Problema do Caixeiro Viajante (PCV)

Para avaliar a performance do RCCA foram realizados testes com os PCV, Ry48p, um PCV assimétrico, de difícil solução, com 48 cidades, o FTV33, um PCV assimétrico, de dificuldade média, com 34 cidades e com o Oliver 30, um PCV simétrico, de fácil solução, com 30 cidades. Esses PCV podem ser encontrados na TSPLIB (REINELT, 2005).

Esses PCV foram escolhidos por possuírem o número de cidades próximo ao número de elementos combustíveis do problema proposto neste trabalho, ou seja, a otimização automatizada da recarga de uma usina nuclear do tipo PWR com simetria de 1/4 e 1/8 de núcleo.

Foram realizados 10 testes, com 10 sementes iniciais diferentes, para cada um dos 3 PCV e calculado o seu valor médio, para apurar a robustez do algoritmo. Sendo que cada teste foi realizado com 1 (modelo serial), 3, 5 e 7 ilhas, e estes mesmos testes foram repetidos mais 3 vezes, uma sem considerar a cooperação dos agentes (sem levar em consideração a matriz Feromônio), outra sem considerar a heurística local e a outra fazendo-se o parâmetro q0 = 0, ou seja, somente considerando as probabilidades da roleta. Foram realizadas 160 execuções de cada um dos PCV selecionados. O número de formigas (agentes) foi considerado sempre igual ao número de cidades do PCV,

sendo que este número pode variar de 1 até o número de cidades existentes no PCV. Pois quanto mais formigas estiverem procurando a solução, mais fácil será para achá-la. Ressalta-se, ainda, que cada formiga foi posicionada inicialmente em uma cidade.

Para a execução do algoritmo RCCA é necessário realizar a escolha dos valores de diversos parâmetros. Cada um desses parâmetros tem um significado no processo de evolução do algoritmo. Nos testes realizados, os valores adotados têm por base os valores utilizados nos testes das referências. Um estudo sobre os mesmos pode ser verificado em GAMBARDELLA e DORIGO, 1996.

Para todos os 3 PCV escolhidos foram utilizados os seguintes parâmetros:

Semente inicial variando de 1 a 10, este parâmetro é um número inteiro longo que nos fornece uma estimativa inicial para as soluções do algoritmo.

Número de gerações = 2000, este parâmetro nos diz o número de vezes que o algoritmo é executado, sendo que o número de iterações é igual ao número de formigas - *n* - multiplicado pelo número de gerações.

Beta = 0 ou 2, este parâmetro é a importância da heurística relativamente ao feromônio (que neste tipo de algoritmo é usualmente 1, salvo em um dos casos teste onde foi utilizado o valor 0).

q0 = 0,9 ou 0,0, nos diz o quanto a busca por soluções vai ser baseada no conhecimento adquirido pelo algoritmo e pela roleta utilizada (se for 0,9, 90% conhecimento adquirido e 10 % roleta).

Alfa = 0,1, parâmetro de evaporação do feromônio.

Número de ilhas = 1, 3, 5, 7, número de algoritmos independentes que são executados independentes.

Intervalo de migração = 10, número de gerações nos quais há interações entre as ilhas (algoritmos).

4.1 Resultados obtidos pela aplicação do RCCA ao PCV Oliver 30

A Tabela 4.1 nos mostra os resultados do PCV Oliver 30 com o parâmetro q0 = 0, ou seja, apenas utilizando a parte probabilística da Equação 2.4. Percebe-se que mesmo utilizando-se a heurística e a cooperação dos agentes, não foi atingido o valor ótimo em nenhuma das execuções do algoritmo, apesar de ser um PCV de relativa facilidade de solução.

PCV Oliver 30	Valor médio	Geração média	Melhor valor	Valor ótimo
1 ilha	462,79	753,3	435,84	423,74
3 ilhas	455,40	719,7	438,99	423,74
5 ilhas	453,44	993,2	444,82	423,74
7 ilhas	447,39	1092,9	440,23	423,74

Tabela 4.1 – Resultados do PCV Oliver 30 com q0 = 0

A Tabela 4.2 nos mostra os resultados do PCV Oliver 30 sem a cooperação dos agentes, ou seja, com a importância do feromônio igual a zero. Percebe-se que mesmo utilizando-se da heurística, não foi atingido o valor ótimo em nenhuma das execuções do algoritmo, apesar de ser um PCV de relativa facilidade de solução.

PCV Oliver 30	Valor médio	Geração média	Melhor Valor	Valor ótimo
1 ilha	437,64	947	435,70	423,74
3 ilhas	435,39	1016	432,18	423,74
5 ilhas	434,85	862.3	432,18	423,74
7 ilhas	433,76	1296	431,90	423,74

Tabela 4.2 – Resultados do PCV Oliver 30 com importância do Fe = 0

A Tabela 4.3 nos mostra os resultados do PCV Oliver 30 com o parâmetro Beta = 0, ou seja, sem a utilização do conhecimento prévio do problema (heurística). Percebe-se que, neste caso, a cooperação dos agentes foi de importância relevante, pois mesmo sem se utilizar a heurística, o algoritmo conseguiu atingir o valor ótimo em todas as tentativas.

PCV Oliver 30	Valor médio	Geração média	Melhor Valor	Valor ótimo
1 ilha	423,91	786,3	423,74	423,74
3 ilhas	423,74	442,5	423,74	423,74
5 ilhas	423,82	201,2	423,74	423,74
7 ilhas	423,74	230,1	423,74	423,74

Tabela 4.3 – Resultados do PCV Oliver 30 com importância de He = 0

A Tabela 4.4 nos mostra os resultados do PCV Oliver 30 com todos os parâmetros ativados de forma otimizada. Percebe-se que, a cooperação dos agentes e a heurística funcionando de forma conjunta colaboraram para uma diminuição do número de gerações para se atingir o valor ótimo. Nota-se que o ótimo foi obtido para todos os números de ilhas.

PCV Oliver 30	Valor médio	Geração média	Melhor valor	Valor ótimo
1 ilha	426,86	505,6	423,74	423,74
3 ilhas	425,30	402,1	423,74	423,74
5 ilhas	423,74	205,7	423,74	423,74
7 ilhas	425,25	109,1	423,74	423,74

Tabela 4.4 – Resultados do PCV Oliver 30

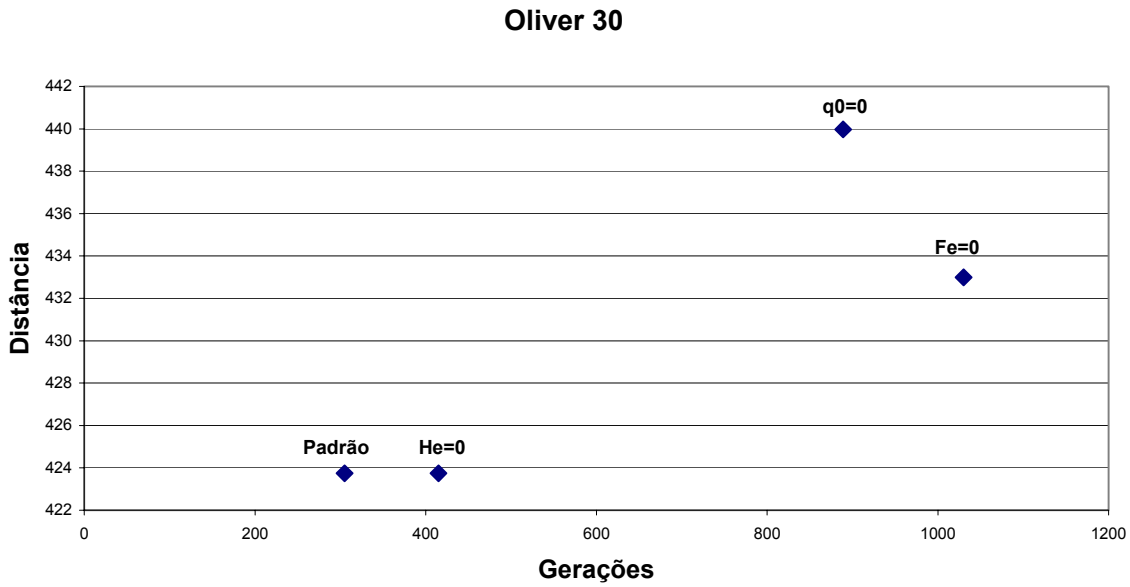

Figura 4.1 – Resultados médios do Oliver 30

Na Figura 4.1 e nas outras similares que serão mostradas, a qualidade do algoritmo dá-se pela proximidade do ponto ao vértice inferior esquerdo, pois quanto mais o ponto se aproximar deste vértice, mais o algoritmo vai ser preciso e rápido e podemos perceber claramente que o ponto chamado de padrão é o que mais se aproxima do ideal, e este ponto é o algoritmo utilizando a heurística e a cooperação dos agentes, com seus valores médios.

Observando os resultados podemos perceber que:

Para o PCV Oliver 30, o mais fácil, tivemos o seguinte comportamento do algoritmo.

- No teste onde o valor q0 foi feito igual a zero, Tabela 4.1, temos que, quanto mais ilhas se utilizaram, mais os valores médios caíram; porém o número de gerações médio aumentou devido ao aumento de diversidade gerado pelas ilhas. O valor ótimo não foi alcançado em nenhum momento, provando que o parâmetro q0 é um parâmetro de

grande importância para o algoritmo, pois sem ele o algoritmo fica totalmente probabilístico.

- No teste onde a importância do feromônio foi igual a zero, Tabela 4.2, temos que, os melhores valores e os valores médios foram melhores do que o caso anterior, mas assim mesmo não se conseguiu atingir o valor ótimo. Com o aumento do número de ilhas se observou uma melhora nos resultados, porém encontrados em um número de gerações maior. Provando também que este é um parâmetro importante para o algoritmo.
- No teste onde a importância da heurística local foi igual a zero, Tabela 4.3, temos que, os valores médios ficaram bem próximos dos valores ótimos, pois este parâmetro, neste caso, não foi essencial ao algoritmo, porém se bem otimizado encontra boas soluções em menores tempos. Observamos também que o valor ótimo foi encontrado com o uso de 1, 3, 5 e 7 ilhas. O número de gerações diminuiu quando aumentamos o número de ilhas.
- No teste onde todos os parâmetros foram utilizados, Tabela 4.4, temos que, os valores médios se encontram todos próximos dos valores ótimos. Em 1, 3 , 5 e 7 ilhas foi encontrado o valor ótimo e o número de gerações médio para se encontrar o valor ótimo foi o menor dos 4 testes.
- E finalmente, no gráfico 4.1, notamos que o ponto que mais se aproxima da situação ótima, ou sejam melhores valores com o mínimo de gerações, é o ponto que corresponde ao algoritmo funcionando com todos os seus parâmetros, que foi chamado de padrão.

4.2 Resultados obtidos pela aplicação do RCCA ao PCV FTV 33

A Tabela 4.5 mostra os resultados do PCV FTV 33, de dificuldade média, com o parâmetro q0 = 0, ou seja, apenas utilizando a parte probabilística da Equação 2.4. Percebe-se que mesmo utilizando-se a heurística e da cooperação dos agentes, não foi atingido o valor ótimo em nenhuma das execuções do algoritmo.

PCV FTV33	Valor médio	Geração média	Melhor valor	Valor ótimo
1 ilha	1647,2	620,1	1576	1286
3 ilhas	1581,7	1112,4	1412	1286
5 ilhas	1550,4	973,8	1498	1286
7 ilhas	1553,5	1039,2	1475	1286

Tabela 4.5 - Resultados do PCV FTV 33 com q0 = 0

A Tabela 4.6 mostra os resultados do PCV FT V33 sem a cooperação dos agentes, ou seja, com a importância do feromônio igual a zero. Percebe-se que, mesmo utilizando-se da heurística não foi atingido o valor ótimo em nenhuma das tentativas.

PCV FTV33	Valor médio	Geração média	Melhor Valor	Valor ótimo
1 ilha	1374,6	876	1329	1286
3 ilhas	1352,8	790	1329	1286
5 ilhas	1346,4	972	1316	1286
7 ilhas	1342,9	908	1316	1286

Tabela 4.6 - Resultados do PCV FTV 33 com a importância de Fe = 0

A Tabela 4.7 mostra os resultados do PCV FT V33 com o parâmetro Beta = 0, ou seja, sem a utilização do conhecimento prévio do problema (Heurística). Percebe-se que, mesmo utilizando-se a cooperação dos agentes não foi atingido o valor ótimo em nenhuma das tentativas.

PCV FTV33	Valor médio	Geração média	Melhor Valor	Valor ótimo
1 ilha	1548,1	822,5	1487	1286
3 ilhas	1524,3	1000,0	1467	1286
5 ilhas	1521,2	1117,7	1450	1286
7 ilhas	1504,1	1208,1	1436	1286

Tabela 4.7 - Resultados do PCV FTV 33 com He = 0

A Tabela 4.8 mostra os resultados do PCV FTV 33 com todos os parâmetros ativados de forma otimizada. Percebe-se que, a cooperação dos agentes e a heurística funcionando de forma conjunta colaboraram para uma diminuição do número de gerações e para o alcance do valor ótimo. Nota-se que o ótimo foi atingido para todos os números de ilhas.

PCV FTV33	Valor médio	Geração média	Melhor valor	Valor ótimo
1 ilha	1299,6	477,1	1286	1286
3 ilhas	1299,5	156,3	1286	1286
5 ilhas	1300,8	688,2	1286	1286
7 ilhas	1292,0	271,9	1286	1286

Tabela 4.8 - Resultados do PCV FTV 33

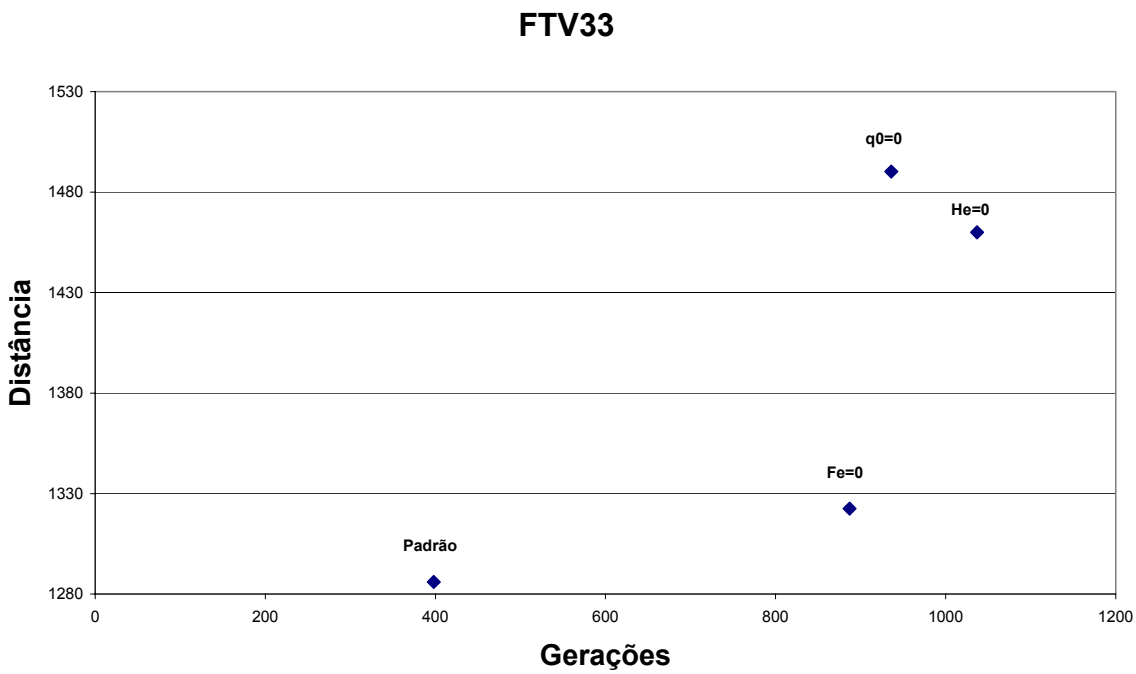

Figura 4.2 – Resultados médios do FTV 33

Para o PCV FTV 33, considerado de dificuldade média, tivemos o seguinte comportamento do algoritmo.

- No teste onde o valor q0 foi feito igual a zero, Tabela 4.5, temos que, quando aumentou-se o número de ilhas os valores médios demonstraram uma queda, porém o número de gerações médio demonstrou um acréscimo, devido ao aumento de diversidade gerado pelas ilhas. O valor ótimo não foi alcançado em nenhum momento, devido à utilização da roleta em todas as gerações para se encontrar qual a próxima cidade a ser visitada, ou seja, com o parâmetro q0 = 0, temos um algoritmo totalmente probabilístico.
- No teste onde a importância do feromônio foi igual a zero, Tabela 4.6, temos que, os melhores valores e os valores médios foram melhores do que o caso anterior, mas assim mesmo não se conseguiu atingir o valor ótimo. Com o aumento do número de ilhas se observou uma melhora

nos resultados, encontrados em um número de gerações similar. Provando também que este é um parâmetro importante para o algoritmo.

- No teste onde a importância da heurística local foi igual a zero, Tabela 4.7, temos que, os valores médios foram piores do que o caso anterior, provando que, neste PCV, a heurística tem mais importância do que no PCV anterior, pois este é de maior dificuldade. Observamos também que os valores foram melhorando à medida que se aumentou o número de ilhas e o número de gerações permaneceu praticamente constante quando aumentamos o número de ilhas.
- No teste onde todos os parâmetros foram utilizados, Tabela 4.8, temos que, os valores médios se encontram todos próximos dos valores ótimos. Em 1, 3 , 5 e 7 ilhas foi encontrado o valor ótimo e o número de gerações médio para se encontrar o valor ótimo foi o menor dos 4 testes.
- E finalmente, no gráfico 4.2, notamos que o ponto que mais se aproxima da situação ótima, ou sejam melhores valores com o mínimo de gerações, é o ponto que corresponde ao algoritmo funcionando com todos os seus parâmetros.

4.3 Resultados obtidos pela aplicação do RCCA ao PCV Rykel 48

A Tabela 4.9 nos mostra os resultados do PCV Rykel 48, de grande dificuldade, com o parâmetro q0 = 0, ou seja, apenas utilizando a parte probabilística da Equação

2.4. Percebe-se que mesmo utilizando-se da heurística e da cooperação dos agentes, não foi atingido o valor ótimo em nenhuma das execuções do algoritmo.

PCV Rykel 48	Valor médio	Geração média	Melhor valor	Valor ótimo
1 ilha	19408,6	1237,1	18856	14422
3 ilhas	18965,3	848,7	18607	14422
5 ilhas	18275,4	975,5	17429	14422
7 ilhas	18415,7	896,6	18143	14422

Tabela 4.9 - Resultados do PCV Rykel 48 com q0 = 0

A Tabela 4.10 mostra os resultados do PCV Rykel 48 sem a cooperação dos agentes, ou seja, com a importância do feromônio igual a zero. Percebe-se que, mesmo utilizando-se da heurística não foi atingido o valor ótimo em nenhuma das tentativas.

PCV Rykel 48	Valor médio	Geração média	Melhor Valor	Valor ótimo
1 ilha	15407,8	803	15320	14422
3 ilhas	15364,7	724	15277	14422
5 ilhas	15340,7	1031	15277	14422
7 ilhas	15324,5	965	15230	14422

Tabela 4.10 - Resultados do PCV Rykel 48 com Fe = 0

A Tabela 4.11 mostra os resultados do PCV Rykel 48 com o parâmetro Beta = 0, ou seja, sem a utilização do conhecimento prévio do problema (Heurística). Percebe-se que, mesmo utilizando-se a cooperação dos agentes não foi atingido o valor ótimo em nenhuma das tentativas.

PCV Rykel 48	Valor médio	Geração média	Melhor Valor	Valor ótimo
1 ilha	19531,1	1035,1	18177	14422
3 ilhas	19024,5	1050,4	18568	14422
5 ilhas	19116,9	1396,5	18594	14422
7 ilhas	18837,2	1269,2	18503	14422

Tabela 4.11 - Resultados do PCV Rykel 48 com He = 0

A tabela 4.12 nos mostra os resultados do Rykel 48 com todos os parâmetros ativados de forma otimizada. Percebe-se que, a cooperação dos agentes e a heurística funcionando de forma conjunta colaboraram para uma diminuição do número de gerações e para o alcance do valor ótimo. Nota-se que o ótimo só foi atingido para o caso de 7 ilhas.

PCV Rykel 48	Valor médio	Geração média	Melhor valor	Valor ótimo
1 ilha	14680,2	838,6	14519	14422
3 ilhas	14610,4	628,3	14495	14422
5 ilhas	14520,8	768,5	14459	14422
7 ilhas	14572,3	564,6	14422	14422

Tabela 4.12 - Resultados do PCV Rykel 48

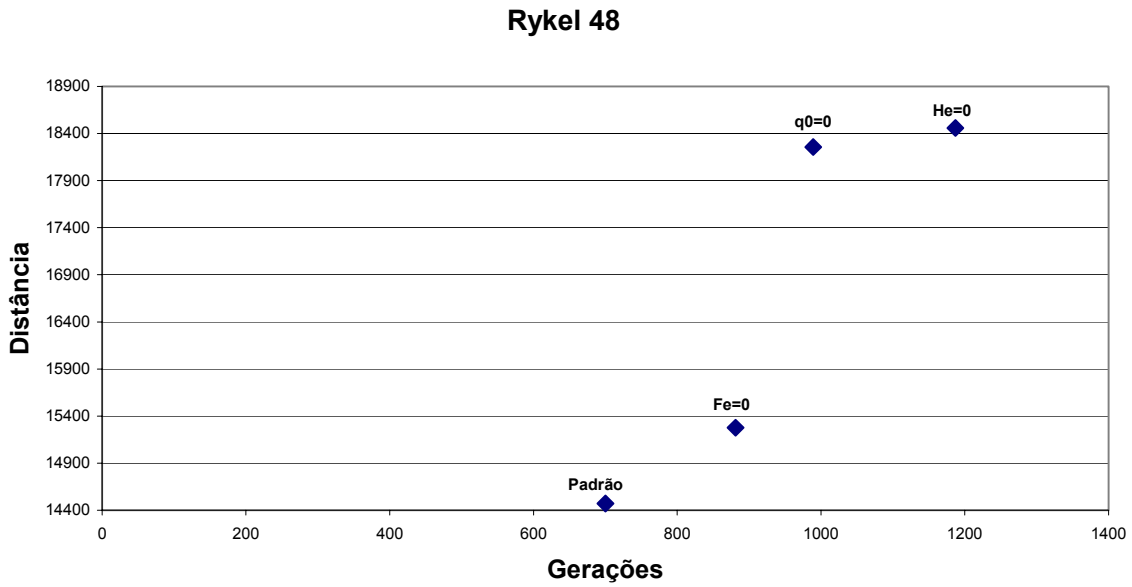

Figura 4.3 – Resultados médios do Rykel 48

Para o PCV Rykel 48, considerado de grande dificuldade, tivemos o seguinte comportamento do algoritmo.

- No teste onde o valor q0 foi feito igual a zero, Tabela 4.9, percebe-se que, por se tratar de um PCV de grande dificuldade o aumento do número de ilhas teve uma pequena influência nos valores médios e melhores valores e o número de gerações médio para se encontrar estes valores não teve grande variação. Porém o valor ótimo não foi alcançado em nenhum momento, devido à utilização da roleta em todas as gerações para se encontrar qual a próxima cidade a ser visitada e devido à dificuldade do PCV.
- No teste onde a importância do feromônio foi igual a zero, Tabela 4.10, temos que, os melhores valores e os valores médios foram melhores do que o caso anterior, mas assim mesmo não se conseguiu atingir o valor ótimo. Com o aumento do número de ilhas se observou uma pequena melhora nos resultados, encontrados em um número de gerações

similar. No teste onde a importância da heurística local foi igual a zero, Tabela 4.11, temos que, os valores médios e melhores valores foram piores do que o caso anterior, provando que, neste PCV, a heurística tem mais importância do que no PCV Oliver 30, pois este PCV é de maior dificuldade. Observamos também que os valores tiveram uma pequena melhora à medida que se aumentou o número de ilhas e o número de gerações aumentou quando aumentamos o número de ilhas. Em PCV de maiores dificuldades a heurística local se mostrou mais importante do que em PCV de pequenas dificuldades.

- No teste onde todos os parâmetros foram utilizados, Tabela 4.12, temos mais uma vez que, os valores médios se encontram todos próximos dos valores ótimos. Somente com 7 ilhas foi encontrado o valor ótimo e o número de gerações médio para se encontrar o valor ótimo foi o menor dos 4 testes.
- E finalmente, no Gráfico 4.3, notamos que o ponto que mais se aproxima da situação ótima, ou sejam melhores valores com o mínimo de gerações, é o ponto que corresponde ao algoritmo funcionando com todos os seus parâmetros.

Com esses resultados podemos chegar a algumas conclusões para a aplicação do RCCA ao PCV:

- O algoritmo funciona sempre melhor com todos os parâmetros ativos.
- A paralelização do algoritmo melhora os resultados sensivelmente, pois no caso de maior dificuldade o valor ótimo só foi encontrado com o uso de ilhas.

CAPÍTULO 5

5. Modelagem para a aplicação das Redes Conectivas de Colônias Artificiais (RCCA) ao Problema da Recarga Nuclear

Neste capítulo é descrita a modelagem do RCCA para ser aplicada ao problema da recarga nuclear e as diferenças básicas entre este processo e o PCV apresentado no capítulo anterior.

5.1 Introdução

O problema da otimização da recarga de uma usina nuclear está possui uma similaridade com o Problema do Caixeiro Viajante (PCV), uma vez que ambos são problemas combinatórios onde os elementos de um conjunto devem ser ordenados de modo que cada elemento ocupe uma única posição na ordenação.

Enquanto o PCV é definido como a busca pela rota que nos forneça a distância mínima percorrida e que passe por todas as cidades apenas uma vez, e ao final do percurso retorne à cidade de origem, o problema da recarga nuclear é uma busca por configurações de elementos combustíveis no núcleo que otimize, por exemplo, a queima destes elementos, sempre obedecendo aos parâmetros de segurança. Com isso, consigue-se uma maior concentração de boro inicial, implicando um maior número de Dias Efetivos à Plena Potência (DEPP), pois cada DEPP a mais equivale a um ganho de:

Potência nominal da usina em MWe x 24 horas x valor do Mwe em R$, que para Angra 1 equivale a 625 x 24 x 92 = R$ 1.380.000,00 e para Angra 2 equivale a 1.355 x 24 x 92 = R$ 2.991.840,00, em valores atuais.

Desta forma, a analogia com o PCV torna-se direta: as cidades representam os elementos combustíveis e suas respectivas posições e a rota representa uma configuração dos elementos combustíveis no núcleo do reator.

O algoritmo RCCA pode ser adaptado para o problema da recarga através de uma modelagem específica, visto que existem algumas diferenças entre os dois problemas. Estas diferenças são descritas a seguir.

5.2 Diferenças entre o PCV e o problema da recarga nuclear

Existem três diferenças fundamentais entre o problema da recarga nuclear e o PCV.

1 - A primeira é em relação à dimensão dos dois problemas

2 - A segunda é a grande dificuldade em se obter uma função heurística local, relacionada à colocação do elemento combustível *s* ao lado do elemento combustível *r*.

3 - A terceira é em relação à analogia direta entre cidades com elementos combustíveis e suas posições, e rotas com configuração de núcleo.

5.2.1 Diferença 1 : Dimensão

As cidades que compõem o PCV estão contidas em um plano, ou seja, as cidades têm coordenadas x,y. O objetivo deste problema é ligá-las de modo sequencial, formando um ciclo fechado de percurso mínimo, lembrando que a função objetivo neste problema é a soma total das distâncias ente as cidades na ordem em que elas estão apresentadas.

A modelagem do RCCA para este problema é simples, o agente constrói seu caminho deslocando-se de cidade em cidade até completar um ciclo fechado. O formato da solução é uma lista contendo a ordem em que as cidades são visitadas. Um exemplo pode ser visto na Figura 5.1, onde é apresentada uma das possíveis rotas para o PCV simétrico Oliver 30.

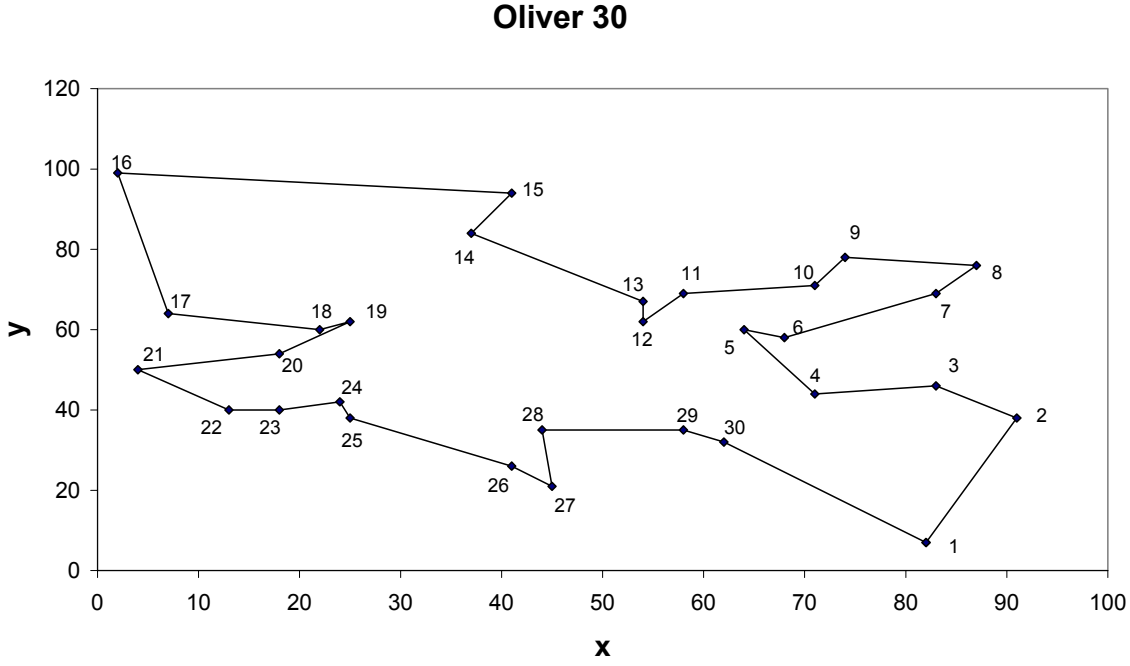

Figura 5.1 – Possível solução do PCV Oliver 30

A rota, ou solução, encontrada na Figura 5.1 pode ser representada como:

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30,

significando que partindo da cidade 1, a próxima cidade a ser visitada é a cidade 2, seguindo-se pelas cidades 3, 4, 5 e assim sucessivamente até voltar à cidade 1.

O problema da recarga nuclear consiste em posicionar os elementos combustíveis no núcleo do reator de modo a se otimizar a função objetivo. A função objetivo pode ser a minimização do fator de pico de potência radial, para uma estratégia de carregamento *out-in*, ou a maximização da concentração crítica de boro solúvel, se o interesse for obter um carregamento tipo baixa-fuga, levando-se sempre em consideração o valor limite do fator de pico de potência radial para a operação da usina nuclear, o qual é estabelecido pelas Especificações Técnicas.

Neste problema, cada agente escolhe um elemento combustível para ocupar uma determinada posição no núcleo do reator. Essas escolhas são feitas até quando não existirem mais elementos combustíveis disponíveis.

Para se manter o mesmo formato da solução gerada pelo RCCA para o PCV, ou seja, uma lista contendo os elementos combustíveis que são escolhidos um a um, foi necessária a construção de um mapa de posicionamento para localizar os elementos combustíveis no núcleo do reator conforme esses elementos fossem sendo escolhidos pelos agentes. Este mapa de posicionamento foi construído de modo a favorecer a atuação de uma heurística local, visto que para a escolha dos elementos combustíveis é necessária a avaliação de uma heurística local e esta escolha se dá sem que se tenha conhecimento da configuração como um todo.

O mapa de posicionamento procura fazer com que dois elementos combustíveis, que sejam escolhidos consecutivamente pelos agentes, ocupem posições no núcleo do reator de modo que ao menos um de seus lados faça fronteira entre si. Assim, a questão da proximidade entre os elementos combustíveis, regidas por alguma heurística local, fica garantida.

A Figura 5.2 mostra um exemplo de mapa de ordenamento. O reator usado neste exemplo foi um reator *PWR* de 121 elementos combustíveis, tal qual Angra 1, utilizando-se uma simetria de 1/8 de núcleo. Os números mostram a posição que os elementos combustíveis ocuparão ao serem escolhidos pelos agentes. Assim, o primeiro elemento combustível escolhido ocupará a posição 1, o segundo elemento combustível escolhido ocupará a posição 2 e assim sucessivamente.

C				
6	7			
5	16	8		
4	17	15	9	
3	18	14	11	10
2	19	13	12	
1	20			

Figura 5.2 – Exemplo de um mapa de ordenamento para um reator com 121 EC e simetria 1/8

Além do mapa de ordenamento, é necessária a inclusão de uma variável nas equações que guiam o RCCA. Essa necessidade origina-se do fato de que dois

elementos combustíveis escolhidos seguidamente contribuem de forma diferente para o resultado final da configuração, dependendo das posições ocupadas por esses elementos combustíveis no núcleo do reator.

No PCV a contribuição para a solução final gerada pelo fato de ir da cidade *r* para a cidade *s* é independente do ponto do caminho em que essa transição ocorre, visto que a contribuição de se ir da cidade *r* para a cidade *s* é sempre igual à distância entre essas duas cidades. Então temos que o custo associado ao arco (*r,s*) independe da posição que ele ocupe na rota.

Porém, na otimização da recarga nuclear isto não ocorre. A contribuição para a solução final dada pelo fato do elemento combustível *s* ser precedido pelo elemento combustível *r* depende das posições desses elementos combustíveis no núcleo do reator, isto é, a contribuição que dois elementos combustíveis seguidos darão para a configuração final se eles estiverem ocupando posições mais internas no núcleo do reator é diferente da contribuição dada pelos mesmos dois elementos combustíveis quando ocupam posições periféricas do núcleo do reator. Assim, o custo associado ao arco (*r,s*) é dependente da posição que este arco ocupe na rota (configuração do núcleo) e dessa forma, o feromônio associado a um arco (*r,s*) nessas condições não representa corretamente a informação de qualidade que este deve expressar.

Deste modo, para incorporar esta característica do problema na modelagem, torna-se necessária a inclusão de uma nova variável nas equações que regem o algoritmo RCCA. Essa nova variável representa a posição do elemento combustível *r* no núcleo do reator e, assim, os arcos (*r,s*) transformam-se nos arcos (*r,s,p*) e o problema

da recarga nuclear passa a ter uma dimensão a mais do que o PCV, e vai representar as posições do núcleo como se fossem níveis de profundidade, representados por planos na Figura 5.3. Dessa forma, ao nos movermos do elemento combustível *r* para o elemento combustível *s* na posição p, a matriz feromônio vai ser atualizada de forma diferente que se movermos do elemento combustível *r* para o elemento combustível *s* na posição p+1.

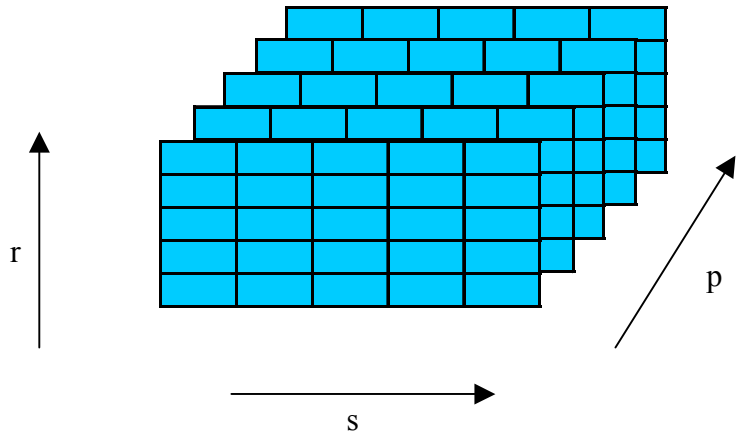

Figura 5.3 – Representação gráfica da variável p

Essas duas modificações na modelagem do problema não afetam o funcionamento do algoritmo RCCA, assim a escolha dos elementos combustíveis para a construção de uma configuração ocorrendo através de uma Regra de Transição de Estado, como no caso do PCV, porém, as equações terão uma dimensão a mais, conforme a Equação (5.1) a seguir.

$$s = \begin{cases} \max \{[FE(r,s,p)]^{\delta} \; x [HE(r,s,p)]^{\beta}\} & se \;\; q \le q_0 \\ Roleta & se \;\; q > q_0 \end{cases} \qquad (5.1)$$

onde Roleta é dado pela distribuição de probabilidades pseudo-randômica-proporcional, mostrada na Equação (5.2).

$$\text{Roleta} = \begin{cases} \dfrac{[FE(r,s,p)]^{\delta} \; x [HE(r,s,p)]^{\beta}}{\sum_{z \in J_k(r)} [FE(r,z,p)]^{\delta} \; x [HE(r,z,p)]^{\beta}} & se \quad s \in J_k(r) \\ 0 & se \quad s \notin J_k(r) \end{cases} \tag{5.2}$$

A regra de atualização local será dada pela Equação (5.3):

$$FE(r, s, p) = (1 - \rho) * FE(r, s, p) + \rho * FEzero \tag{5.3}$$

A regra de atualização global será dada pela Equação (5.4):

$$FE(r, s, p) = (1 - \alpha) * FE(r, s, p) + \alpha * (W / \text{melhor resultado}) \tag{5.4}$$

Após concluída a construção de todas as configurações, cada uma delas será submetida a um código de Física de Reatores que calculará o valor das grandezas necessárias à otimização.

Para a construção destas configurações é necessária uma modificação no algoritmo. Devido às simetrias do núcleo, elementos que ocupavam posições de quartetos no ciclo anterior devem continuar ocupando posições de quarteto e elementos que ocupavam posições de octetos no ciclo anterior devem continuar ocupando posições de octeto. Devido a esta exigência, elementos de quarteto só podem ocupar as posições 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 e elementos de octeto só podem ocupar as posições 11, 12, 13, 14, 15, 16, 17, 18, 19, 20 como podemos ver na Figura 5.2. Com essas limitações o número de combinações cai de 20! para 10! x 10!, ou seja, com isso o espaço de busca é reduzido em 184.756 vezes.

Pode-se ter interesse em uma estratégia de carregamento nuclear do tipo *out-in* ou tipo baixa-fuga. Dessa forma, para cada umas das estratégias de carregamento supracitadas usa-se um cálculo diferente para a função objetivo.

Para uma estratégia do tipo *out-in* a função objetivo utilizada foi a de minimização do valor do pico de potência radial, calculado pelo código de Física de Reatores.

Para uma estratégia do tipo baixa-fuga a função objetivo utilizada foi a seguinte:

Se o fator de pico de potência radial for maior do que o limite de especificação técnica, 1,435 no caso de Angra 1, a função objetivo é o próprio fator de pico de potência radial.

Se o fator de pico de potência radial for menor do que o limite de especificação técnica, a função objetivo é Fator Aptidão / Boro, onde Fator Aptidão é um valor empírico que neste trabalho recebeu o valor 1.

Expressando-se em lógica computacional para minimização da função objetivo, temos:

```
Se fator pico > valor limite então
        Função objetivo = fator pico
Senão
        Função objetivo = Fator Aptidão / Boro
Fim do se
```

Esta função nos fornece exatamente o que queremos, ou seja, para valores de pico de potência radial acima do valor limite da especificação técnica, queremos minimizá-los, pois com estes valores acima do limite esta configuração de núcleo torna-

se inválida, e para valores abaixo deste limite, onde as configurações de núcleo tornam-se válidas, queremos maximizar a concentração crítica de boro solúvel. Com o uso desta função percebemos claramente que ao alcançarmos o valor limite do pico de potência radial da especificação técnica (1,435), o valor de boro tende a ficar estável ou subir, e o valor do pico de potência radial oscila abaixo de 1,435.

5.2.2 – Diferença 2 : Heurística Local

Muitos algoritmos inteligentes desenvolvidos para otimização, como por exemplo o Algoritmo Genético, são guiados na direção das boas soluções por heurísticas utilizadas para avaliar as soluções geradas. Essas heurísticas levam em consideração os conhecimentos relativos ao problema como um todo e, por isso, são denominadas heurísticas globais.

O algoritmo RCCA, além de utilizar uma heurística global, representada pelo reforço de aprendizagem, também faz uso de uma heurística local. A heurística local leva em consideração o conhecimento de apenas uma parte do problema, o que simplifica sua aplicação em relação à heurística global.

A heurística local procura expressar a relação existente entre dois elementos quaisquer do conjunto de elementos do problema, para que essa relação conduza a uma melhora da solução final. Em outras palavras, através de um conhecimento que englobe um pequeno número de elementos que estão próximos na ordenação, pretende-se obter uma melhoria na solução global. Assim, para conseguir uma boa heurística local a

questão é: escolhido um determinado elemento, qual será o melhor elemento, dentre os possíveis, para ser usado como próximo elemento na sequência de ordenação ?

A complexidade da heurística local é inteiramente dependente do problema a ser tratado. Pode-se usar uma heurística simples como a heurística do vizinho mais próximo, que é usada no PCV ou pode ser necessária a inclusão de termos referentes à informação de todos os elementos do conjunto utilizado no problema de otimização.

No caso da otimização da recarga nuclear, encontrar uma heurística local satisfatória não é uma tarefa simples, pois esta heurística é referente ao processo de queima dos elementos combustíveis durante a produção de energia e trata-se de um processo de grande complexidade envolvendo grandezas não lineares.

Conceitualmente, deve-se procurar uma heurística local capaz de preencher o mapa de ordenamento dos elementos combustíveis de modo a conseguir configurações que levem a fatores de pico de potência radial com valores menores do que o limite de especificação técnica em tempos razoavelmente aceitáveis, ou seja, obter boas configurações de núcleo em tempos menores dos que os obtidos sem o auxílio da heurística local.

5.2.3 – Diferença 3 : Analogia entre cidades com elementos combustíveis e suas posições, e rotas com configurações de núcleo

Quando fazemos a analogia entre o PCV e o problema da recarga nos vem diretamente a relação de cidades e elementos combustíveis e rotas com configurações de núcleo, ou seja, um problema com 20 elementos combustíveis é equivalente a um PCV com 20 cidades? Esta afirmação pode estar correta se considerarmos um elemento combustível sendo colocado inicialmente em somente uma posição de núcleo, mas isto é uma simplificação do problema e na verdade este elemento combustível pode ser colocado inicialmente em todas as posições do núcleo, salvo em casos de proibições físicas, como por exemplo elementos com veneno queimável em posições de bancos de controle.

Então a nossa analogia ficaria da seguinte forma: se temos 20 elementos combustíveis e estes elementos podem ocupar todas as posições do núcleo (não levando em consideração quartetos e octetos), temos um problema equivalente a 400 cidades, pois, temos para cada posição de núcleo 20 possibilidades iniciais e como temos 20 posições de núcleo, temos um total de 400 possibilidades e um total de 200 cidades se levarmos em consideração as simetrias de quartetos e octetos.

A maioria dos testes foi feita levando-se em consideração a simplificação do problema com apenas 20 formigas representando 20 elementos combustíveis, devido ao tempo exigido para se ter 200 formigas por teste, exceto três testes que foram realizados com o problema completo, um apenas com a heurística global e os outros dois com as duas heurísticas que obtiveram os melhores resultados nos problemas simplificados.

5.3 Heurísticas Utilizadas

Foram testadas seis heurísticas locais, uma levando em consideração a queima dos elementos combustíveis, outras quatro levando em consideração o k-infinito dos elementos combustíveis, uma heurística utilizando conhecimentos adquiridos durante gerações iniciais e duas heurísticas globais, uma utilizando 20 formigas e outra com 200 formigas.

A primeira heurística local procura levar em consideração as informações sobre todos os elementos combustíveis que serão posicionados no núcleo do reator. Seu objetivo é fazer com que a distribuição dos elementos combustíveis leve a uma distribuição de fator de pico de potência mais uniforme, uma vez que ela procura posicionar os elementos combustíveis de modo a manter a média da queima local o mais próximo possível da média da queima global. Esta heurística foi denominada de heurística do nivelamento e sua expressão é dada pela Equação (5.5):

$$HE(r,s) = \frac{1}{\left| \frac{Q(r)+Q(s)}{2} - \frac{\sum_{i=0}^{n^\circ \text{ total de elementos combustíveis}} Q(i)}{n^\circ \text{ total de elementos combustíveis}} \right|} \tag{5.5}$$

A segunda heurística local utilizada procura montar um núcleo formado por elementos combustíveis com k-infinito alto e k-infinito baixo alternadamente, este núcleo é conhecido por “tabuleiro de xadrez”. Seu objetivo é fazer com que a distribuição dos elementos combustíveis leve a uma distribuição de fator de pico de potência radial que não ultrapasse o limite de especificação técnica, pois um elemento

combustível com k-infinito alto tende a aumentar o valor do fator de pico de potência radial, mas ao colocar um elemento combustível com k-infinito baixo ao seu lado, o valor de fator de pico de potência radial tende a diminuir. Esta heurística foi também denominada de heurística do tabuleiro de xadrez, e sua expressão é dada pela Equação (5.6):

$$HE(r,s) = Abs(K\inf(r) - K\inf(s)) \tag{5.6}$$

A terceira heurística local, que na verdade é a primeira heurística, com a substituição da queima por k-infinito, procura levar em consideração as informações sobre todos os elementos combustíveis que serão posicionados no núcleo do reator. Seu objetivo é fazer com que a distribuição dos elementos combustíveis leve a uma distribuição de fator de pico de potência mais uniforme, uma vez que ela procura posicionar os elementos combustíveis de modo a manter a média do k-infinito local o mais próximo possível da média do k-infinito global. Esta heurística, apesar de ser muito parecida com a heurística 1 (Nivelamento), foi testada devido a haver uma diferença grande entre queima e k-infinito, pois podemos ter um elemento com uma queima grande e ainda ser mais reativo do que um elemento novo, já que temos enriquecimentos diferentes. Esta heurística foi denominada de heurística do k-infinito médio, e sua expressão é dada pela Equação (5.7):

$$HE(r,s) = \frac{1}{\left| \frac{K\inf(r) + K\inf(s)}{2} - \frac{\sum_{i=0}^{n^\circ \text{ total de elementos combustíveis}} K\inf(i)}{n^\circ \text{ total de elementos combustíveis}} \right|} \tag{5.7}$$

A quarta heurística local divide o oitavo de núcleo em três regiões, região exterior, região mediana e região central, com mostra a Figura 5.4 e faz a distribuição dos elementos combustíveis da seguinte forma: Nas regiões exterior e mediana são colocados elementos combustíveis com k-infinito próximos e na região central é feita uma distribuição do tipo tabuleiro de xadrez, ou seja, com elementos de k-infinito alto e baixo alternadamente. Esta heurística foi denominada heurística das regiões e suas expressões são dadas pelas Equações (5.8):

```
Se região exterior ou região mediana então
        HE(r,s,p) = 1 / Abs((k_inf(r) - k_inf(s)))
Fim se

Se região central então
        HE(r,s,p) = Abs((k_inf(r) - k_inf(s)))
Fim se                                                    (5.8)
```

Onde *r*, *s* e *p*, são elemento *r*, elemento *s* e *p* posição do núcleo, respectivamente.

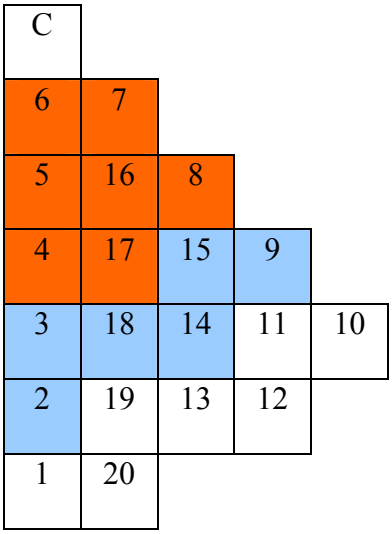

Figura 5.4 – Três regiões do núcleo e C o elemento central

A quinta heurística local divide, novamente, o oitavo de núcleo em três regiões, região exterior, região mediana e região central, com mostra a Figura 5.4 e faz a distribuição dos elementos combustíveis da seguinte forma: Na região exterior são

colocados elementos combustíveis com k-infinito da seguinte forma: se a média, dos k-infinito, dos elementos combustíveis (r,s) for menor do que a média global de todos os elementos combustíveis, a heurística é: valor absoluto de (k_inf(k) - k_inf(l)) multiplicado por um fator 2, senão multiplicado por 1, pois, com isso, queremos aumentar o valor da heurística nestas posições, pois estamos em posição exterior e o k-infinto "resultante" destes dois elementos combustíveis é um valor "baixo", com isso tentamos colocar elementos com k-infinito baixos nesta região. Nas regiões mediana e central são colocados elementos combustíveis com k-infinito da seguinte forma: se a média, dos k-infinito, dos elementos combustíveis (r,s) for maior do que a média global de todos os elementos combustíveis a heurística é: valor absoluto de (k_inf(k) - k_inf(l)) multiplicado por um fator 2, , senão multiplicado por 1, pois, com isso, queremos aumentar o valor da heurística nestas posições, pois estamos em posição de meio e centro de núcleo e o k-infinto "resultante" destes dois elementos combustíveis é um valor "alto", com isso tentamos colocar elementos com k-infinito altos nesta região. Esta heurística foi denominada heurística da baixa-fuga e suas expressões são dadas pelas Equações (5.9):

```
Se região exterior então
        Se (k_inf(r) + k_inf(s)) < 2 * k_inf_Medio então
                HE(r,s,p) = 2 * Abs(k_inf(r) - k_inf(s))
        Senão
                HE(r,s,p) = Abs(k_inf(r) - k_inf(s))
        Fim se
Fim se

Se região mediana ou central então
        Se (k_inf(r) + k_inf(s)) > 2 * k_inf_Medio então
                HE(r,s,p) = 2 * Abs(k_inf(r) - k_inf(s))
        Senão
                HE(r,s,p) = Abs(k_inf(r) - k_inf(s))
        Fim se
Fim se                                                          (5.9)
```

onde *r*, *s* e *p*, são elemento *r*, elemento *s* e *p* posição do núcleo, respectivamente.

A sexta heurística local, que na verdade é uma modificação do algoritmo, “aproveita-se” do aprendizado global das formigas, obtido pela atualização do feromônio, e usa este como heurística da seguinte forma: O algoritmo é iniciado normalmente, com o parâmetro beta, que define a importância da heurística local, com valor zero, depois de um determinado número de gerações, que neste caso em particular foi usado o valor 50, o algoritmo pára sua execução, copia a matriz feromônio, que representa o conhecimento global adquirido até o momento, integralmente para a matriz heurística. O valor de beta é então trocado para um valor positivo, que neste caso em particular foi usado o valor 1, e preenche a matriz feromônio integralmente com FEzero, ou seja, a matriz feromônio volta a ter seus valores iniciais. Então já na geração 51 temos um algoritmo com uma boa heurística, pois o que foi aprendido pelas formigas ao longo das 50 gerações agora se transformou em um conhecimento inicial. Com esta heurística, mesmo sem termos uma real noção do que é “bom” ou “ruim” localmente, como é o caso da recarga, ao escolhermos quais elementos combustíveis serão alocados em quais posições, pois só podemos avaliar o núcleo como um todo e não com apenas algumas regiões preenchidas com elementos combustíveis e outras não. Esta heurística foi denominada heurística Parasita, pela sua forma de atuar, esperando a heurística global achar um “bom“ caminho e depois se aproveitando desta informação para seguir seu próprio caminho.

A sétima heurística testada foi a heurística global, ou seja, sem se utilizar nenhuma heurística local. Este teste foi executado com o parâmetro beta, que define a importância da heurística local, com valor zero. Com isso o algoritmo só se utiliza do conhecimento adquirido ao longo das gerações através da atualização da matriz feromônio.

A oitava, nona e décima heurísticas são as heurísticas completas, onde o número de formigas foi igual a 200, interpretando o problema da uma forma mais correta. A oitava foi a heurística global completa, a nona a Nivelamento completa e a décima a Parasita completa.

Nos testes no decorrer deste trabalho compararemos as 10 heurísticas.

5.4 O Código de Física de Reatores RECNOD

5.4.1 Introdução

Para a avaliação das configurações de núcleo, geradas pelo algoritmo SCF, pode ser usado qualquer código de Física de Reatores existente no mercado. Porém, para poder comparar-se diretamente o RCCA e o Algoritmo Genético (CHAPOT, 2000) foi usado o código nodal RECNOD, que é um código de física de reatores bidimensional, dois grupos de energia, utilizado no trabalho CHAPOT (2000).

5.4.2 O Método de Expansão de Fluxo

No final da década de 50, foi criado o programa de computadores PDQ (CHAPOT, 2000), considerado um marco na análise dos problemas da física de reatores. Esses pesquisadores empregaram o Método de Diferenças Finitas (MDF) (CHAPOT, 2000) para resolver a equação da difusão de nêutrons bidimensional, para poucos grupos de energia. Embora a utilização do MDF tenha sido um grande avanço, a aplicação deste método sofre limitações pela grande necessidade de memória exigida.

Conforme Claro (CLARO, 1992), quando aplicado a cálculos bidimensionais, por exemplo, o MDF requer espaçamento de malha da ordem de 2 a 3 cm para modelar reatores a água pressurizada. Isto implica a necessidade de computar o fluxo de nêutrons em mais de 1 milhão de pontos, para cada grupo de energia. Os altos custos computacionais do MDF levaram à criação de uma metodologia de cálculo menos rigorosa, porém mais eficiente: os *métodos nodais*. Esses métodos partem do pressuposto de que o núcleo do reator possa ser decomposto em sub-regiões, relativamente grandes, chamadas *nodos*. Cada nodo pode, por exemplo, ter a largura de um elemento combustível (cerca de 20 cm). Por esta razão, os métodos nodais são denominados métodos de malha grossa. No caso de estudos bidimensionais, em geometria cartesiana, utiliza-se nodos retangulares ou quadrados. Em análises tridimensionais usa-se nodos com a forma de paralelepípedos retângulos. A técnica dos métodos nodais está calcada em duas hipóteses:

- Os parâmetros nucleares são supostos uniformes dentro de um nodo;
- Em cada nodo o fluxo pode ser representado por uma expansão em funções polinomiais, para os métodos que usam expansão polinomial.

No final dos anos 70, Langenbuch *et al.* (LANGENBUCH et al, 1977) criaram um método, denominado Método de Expansão de Fluxo (FEM). Tendo por base o FEM, Montagnini *et al.* (MONTAGNINI et al, 1994) desenvolveram um método nodal, que, segundo esses autores, é capaz de produzir erros máximos na distribuição de potência inferiores a 1,5%.

5.4.3 FEM-M: Um Método De Expansão de Fluxo Alternativo

A diferença fundamental entre o FEM clássico e o modelo alternativo proposto por Montagnini *et al.* (1994), que a partir deste ponto será designado como FEM-M, é que, enquanto o método de Langenbuch *et al.* (LANGENBUCH et al, 1977) calcula fluxos pontuais (no centro e nos pontos médios dos lados do nodo), o FEM-M faz uso de fluxos médios, mostrados na Figura 5.5 e definidos como se segue:

- $\phi^{i,j}$ - fluxo médio na área do nodo;
- $\phi^{i-1/2,j}$ - fluxo médio no lado esquerdo do nodo;
- $\phi^{i+1/2,j}$ - fluxo médio no lado direito do nodo;
- $\phi^{i,j-1/2}$ - fluxo médio no lado inferior do nodo;
- $\phi^{i,j+1/2}$ - fluxo médio no lado superior do nodo.

Da Figura 5.4, tem-se que.

$$x_{i\pm 1/2} = x_i \pm \frac{1}{2} h_{xi} \quad \text{e} \quad y_{j\pm 1/2} = y_j \pm \frac{1}{2} h_{yj}.$$

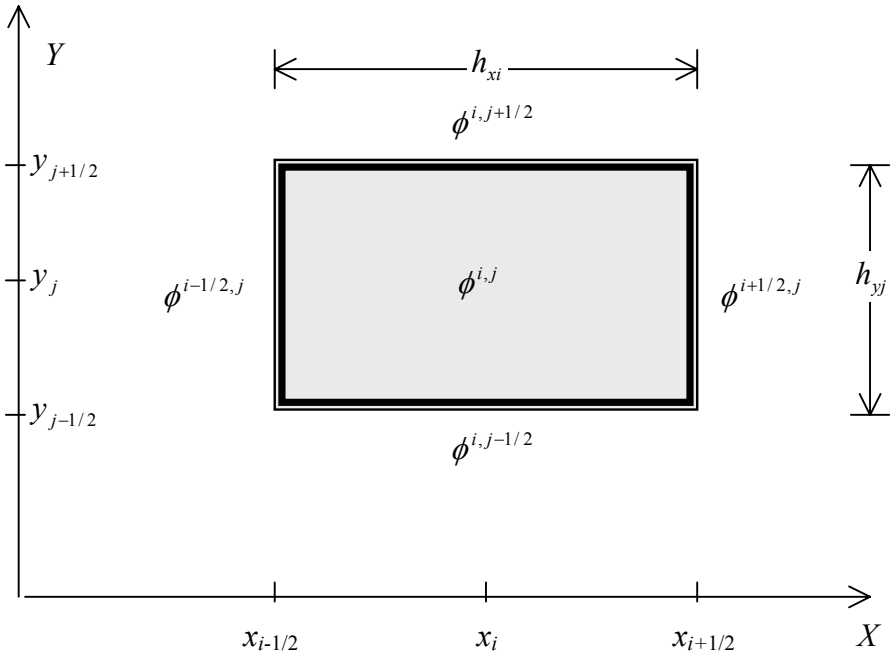

Figura 5.5 Esquema de um Nodo do Método FEM-M

5.4.4 Limitações do Código RECNOD

Um código nodal de física de reatores, de uso comercial para estudos de otimização de recargas de reatores, deve conter os seguintes módulos: *fluxo-potência-reatividade*; *queima de combustível; pesquisa de criticalidade com boro solúvel; modelos de realimentação*, onde se incluem a *correção da densidade do moderador* e a *realimentação Doppler*, entre outros, e a *reconstrução da distribuição de densidade de potência pino a pino*, para que se possa determinar o fator de pico de potência radial (F_{XY}) das varetas combustíveis.

O código RECNOD não possui os módulos com os modelos de realimentação termohidráulica e nem aquele para a reconstrução da distribuição de potência pino a pino, necessário para o cálculo do fator de pico de potência radial. Por isso, o fator de pico de potência radial foi substituído pela máxima potência média relativa, parâmetro este, que segundo CHAPOT (2000), pode substituir o fator de pico de potência radial com erro de ± 2%. Além do mais, como nosso foco é o algoritmo de otimização, a variação deste parâmetro equivale em termos computacionais ao comportamento não linear do fator de pico de potência radial.

CAPÍTULO 6

6. Resultados obtidos pela aplicação das Redes Conectivas de Colônias Artificiais (RCCA) ao Problema da Recarga Nuclear com auxílio do Código RECNOD

Como o objetivo principal deste trabalho de pesquisa é propor uma nova técnica utilizando-se colônias de formigas para o problema da recarga de reatores nucleares, neste capítulo apresentamos os resultados do sistema RCCA/RECNOD.

O RCCA foi desenvolvido em linguagem de programação Visual Basic 6.0, assim como a interface de comunicação com o código de física de reatores RECNOD, que foi elaborado em linguagem de programação Fortran no trabalho CHAPOT, 2000 e nos foi gentilmente cedido. O ciclo otimizado foi o ciclo 7 da usina nuclear Angra 1.

6.1 Metodologia Utilizada

Foram realizados, no total, 160 testes, os quais foram divididos da seguinte forma. Para cada uma das (7+3) heurísticas escolhidas (7 com o problema simplificado, com 20 formigas e 3 com o problema de forma completa, com 200 formigas), foram feitos 16 testes. Dentre esses 16 testes, foram feitos testes variando-se as ilhas de 1,3,5 e 7 ilhas, com duas funções objetivo diferentes, e com duas variações da quantidade de feromônio inicial. Então temos 4 testes com ilhas, 2 testes com funções objetivo e 2

testes com feromônio inicial, dando um total de 4 x 2 x 2 = 16 testes, ou seja, todas as combinações de testes possíveis.

6.1.1 Funções Objetivo Utilizadas

Foram utilizadas, neste trabalho, duas funções objetivo. Uma para recarga nuclear do tipo *out-in* e uma para recarga nuclear do tipo baixa-fuga. A utilizada para recarga tipo *out-in* é simplesmente a minimização do fator de pico de potência radial, lembrando que neste trabalho foi utilizada a potência média, por limitações do código RECNOD, e a utilizada para recarga do tipo baixa-fuga foi a seguinte:

```
Se potência média > 1,395 então
        Função objetivo = potência média
Senão
        Função objetivo = Fator Aptidão / Boro
Fim do se
```

O limite do fator de pico de potência radial do núcleo é dado pelas Especificações Técnicas da usina Angra 1 e tem o valor de 1,435, mas segundo estudos realizados por CHAPOT, 2000, o valor aconselhável para se utilizar, quando trabalhamos com potência média, ao invés de fator de pico de potência radial, deve ser 1,395. E o valor utilizado da variável Fator Aptidão foi sempre 1.

6.1.2 Ajuste dos parâmetros do Sistema RCCA/RECNOD

Este tipo de algoritmo é muito sensível aos parâmetros escolhidos e a escolha de alguns destes se dá de forma empírica. Como ainda não havíamos feito um número de testes suficientemente grande para haver alguma confiabilidade nos efeitos de cada parâmetro, foi decidido se fazer um ajuste de parâmetros para as ordens de grandeza dos testes feitos em PCV, ou seja, utilizamos os parâmetros ótimos do PCV para o problema da recarga, apenas com um ajuste das ordens de grandeza das funções objetivo. E os parâmetros utilizados foram os seguintes.

Semente inicial = 951968, este parâmetro é um número inteiro longo que nos fornece a montagem aleatória inicial para as soluções do algoritmo.

Número de gerações = 300, este parâmetro nos diz o número de vezes que o algoritmo é executado, sendo que o número de iterações é de n x 300 (n = número de formigas, que nestes testes foi igual a 20 ou 200).

Número de agentes = 20 ou 200, em 7 casos foi utilizado o valor de 20 formigas, exceto nos três casos Completos, onde foram utilizadas 200 formigas.

Beta = 0 ou 1, este parâmetro é a importância da heurística relativamente ao feromônio (que neste tipo de algoritmo é usualmente 1, salvo em dois dos casos teste que foi utilizado o valor 0).

FEzero = 1,34e-7 para o **caso 1** e 1,34e-8 para o **caso 2**. FEzero é a quantidade de feromônio inicial nas trilhas das formigas.

q0 = 0,9, este parâmetro nos diz o quanto a busca por soluções vai ser baseada no conhecimento adquirido pelo algoritmo e pela roleta utilizada, no caso, 0,9 é 90% de conhecimento adquirido e 10 % roleta).

alfa = 0,1, parâmetro de evaporação do feromônio, equivalente à evaporação da quantidade de feromônio nas trilhas das formigas reais.

Número de ilhas = 1, 3, 5, 7, número de algoritmos que são executados de forma independente.

Intervalo de migração = 5, número de gerações nas quais há interações entre as ilhas (algoritmos).

Tipo de migração => Migração do tipo dinâmica, onde cada algoritmo troca informações com diferentes algoritmos (ilhas) em cada migração, como mostrado na Figura 3.5.

6.1.3 Sistema RCCA/RECNOD

A seguir são apresentadas as duas telas do sistema RCCA/RECNOD, onde na Figura 6.1 temos a tela de entrada de dados e na Figura 6.2 temos a tela de execução.

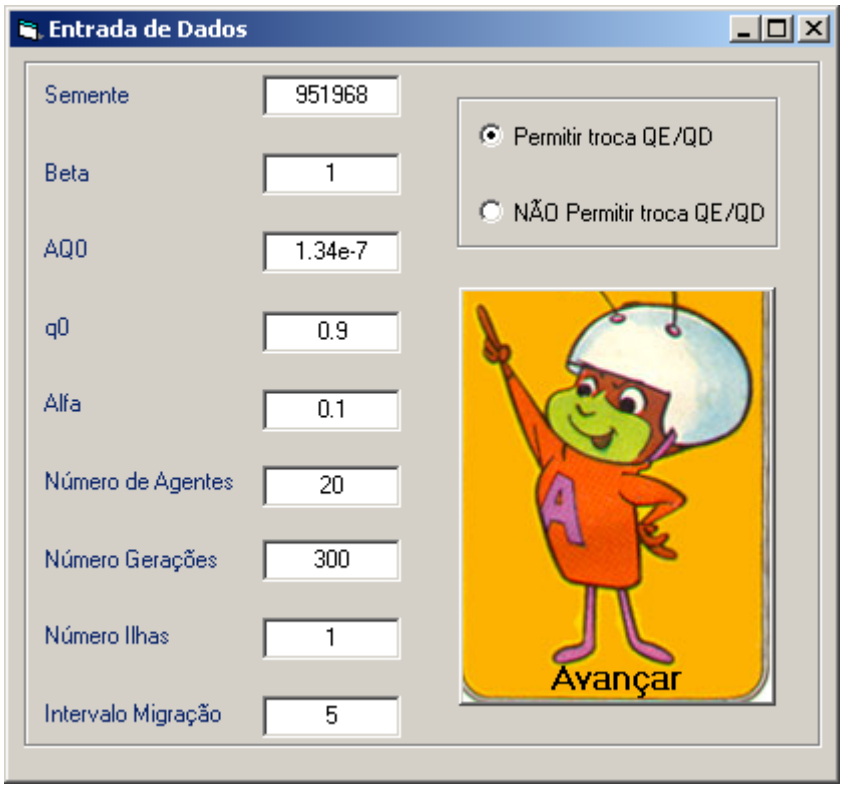

Figura 6.1 – Tela de entrada de dados do sistema RCCA/RECNOD

Nesta tela temos a configuração de todos os parâmetros do sistema e a opção de haver a possibilidade de troca de elementos de quartetos de eixo com quartetos de diagonal. Em todos os testes esta opção foi permitida.

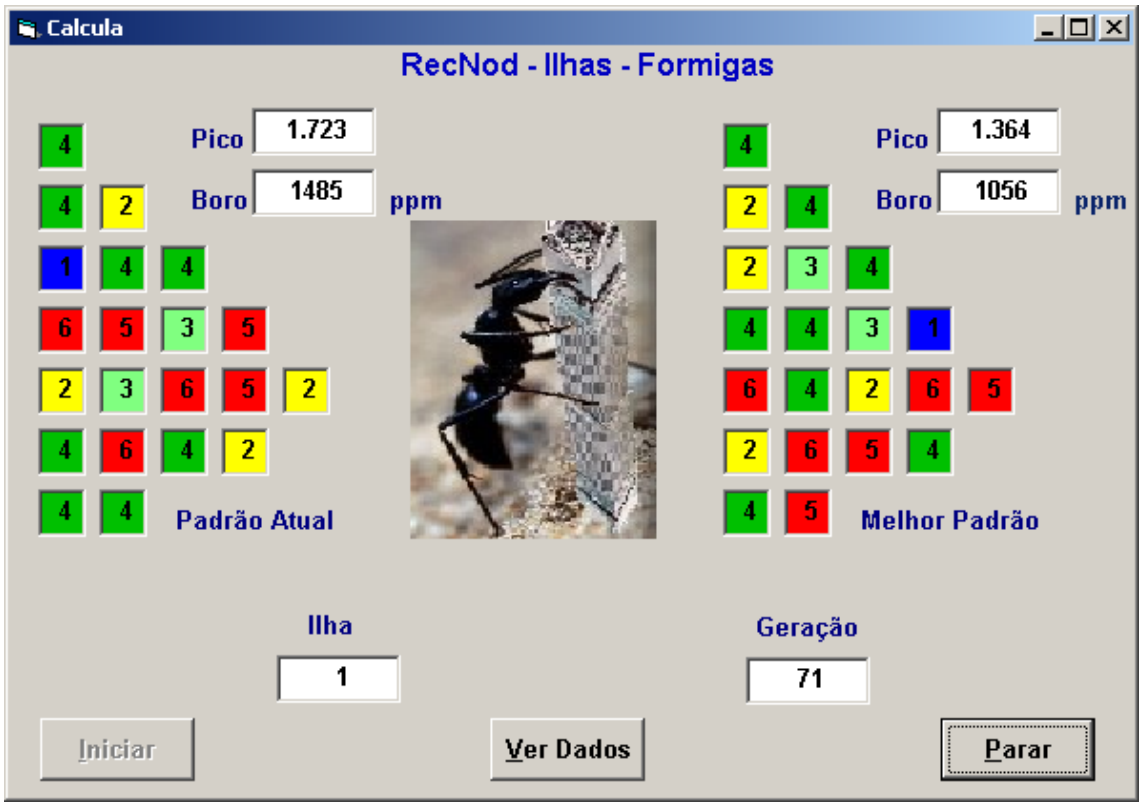

Figura 6.2 – Tela de execução do sistema RCCA/RECNOD

Nesta tela temos, o melhor resultado de todos e o valor atual da potência média e da concentração de boro, assim como a representação de 1/8 de núcleo com a colocação dos elementos nas suas devidas posições. Cada número dentro de um dos quadrados representa um tipo de elemento (os EC de número de 1 a 4 são EC usados e os de número 5 e 6 são novos).

6.2 Resultados Obtidos com o Sistema RCCA/RECNOD

Para a execução do sistema RCCA/RECNOD é necessário se realizar a escolha dos valores de diversos parâmetros empiricamente. Cada um desses parâmetros tem um

significado no processo de evolução do algoritmo. Veremos abaixo os resultados do RCCA com suas heurísticas correspondentes.

6.2.1 Heurística do Nivelamento

A heurística local utilizada, neste caso, foi a de se tentar colocar pares de elementos combustíveis, com a média de suas queimas, o mais próximo possível da queima média de todos os elementos combustíveis que compõem o oitavo de núcleo, de modo a nivelar a potência média. Com isso, tentamos conseguir picos de potência mais baixos e assim ficar dentro do limite de especificação técnica da usina Angra 1.

A Tabela 6.1 nos mostra os resultados da heurística do Nivelamento com a função aptidão para a maximização da concentração de boro.

	Número de Ilhas	**Geração**	**Pot. Média**	**Boro**
Caso 1	1	193	1,382	1148
Caso 2	1	98	1,382	1326
Caso 1	3	71	1,368	1121
Caso 2	3	173	1,380	1211
Caso 1	5	106	1,313	1086
Caso 2	5	93	1,384	1360
Caso 1	7	253	1,320	1106
Caso 2	7	90	1,383	1292
	Valores médios	**134**	**1,364**	**1206**

Tabela 6.1 – Resultados da heurística do Nivelamento para maximização de boro

A Tabela 6.2 nos mostra os resultados da heurística do Nivelamento com a função aptidão para a minimização da potência média.

	Número de Ilhas	**Geração**	**Pot. Média**	**Boro**
Caso 1	1	157	1,319	981
Caso 2	1	170	1,435	1047
Caso 1	3	197	1,350	941
Caso 2	3	296	1,308	895
Caso 1	5	252	1,363	1206
Caso 2	5	180	1,315	889
Caso 1	7	146	1,348	1006
Caso 2	7	105	1,371	874
	Valores médios	**187**	**1,351**	**979**

Tabela 6.2 – Resultados da heurística do Nivelamento para minimização da potência média

6.2.2 Heurística do Tabuleiro de Xadrez

Nesta segunda heurística local utilizada procuramos montar um núcleo formado por elementos combustíveis com k-infinito alto e k-infinito baixo alternadamente. Este núcleo é conhecido por "tabuleiro de xadrez". Seu objetivo é fazer com que a distribuição dos elementos combustíveis leve a uma distribuição de potência média que não ultrapasse o limite de especificação técnica, pois um elemento combustível com k-infinito alto tende a aumentar o valor da potência média, mas ao colocar um elemento

combustível com k-infinito baixo ao seu lado, o valor da potência média tende a diminuir.

A Tabela 6.3 nos mostra os resultados da heurística Tabuleiro de Xadrez com a função aptidão para a maximização da concentração de boro.

	Número de Ilhas	**Geração**	**Pot. Média**	**Boro**
Caso 1	1	184	1,324	1128
Caso 2	1	183	1,371	1052
Caso 1	3	216	1,393	991
Caso 2	3	208	1,395	1073
Caso 1	5	294	1,390	945
Caso 2	5	75	1,385	1120
Caso 1	7	11	1,393	1176
Caso 2	7	155	1,382	1033
	Valores médios	**165**	**1,379**	**1064**

Tabela 6.3 – Resultados da heurística Tabuleiro de Xadrez para maximização de boro

A Tabela 6.4 nos mostra os resultados da heurística Tabuleiro de Xadrez com a função aptidão para a minimização da potência média.

	Número de Ilhas	**Geração**	**Pot. Média**	**Boro**
Caso 1	1	183	1,371	1052
Caso 2	1	187	1,377	1087
Caso 1	3	151	1,384	1035
Caso 2	3	196	1,320	993
Caso 1	5	126	1,346	962
Caso 2	5	31	1,401	1025
Caso 1	7	82	1,326	942
Caso 2	7	301	1,336	929
	Valores médios	**157**	**1,358**	**1003**

Tabela 6.4 – Resultados da heurística Tabuleiro de Xadrez para minimização da potência média

6.2.3 Heurística do K Infinito Médio

Nesta terceira heurística local, na verdade a heurística 1 com a substituição da queima por k-infinito, procuramos levar em consideração as informações sobre todos os elementos combustíveis que serão posicionados no núcleo do reator. Seu objetivo é fazer com que a distribuição dos elementos combustíveis leve a uma distribuição de potência média mais uniforme, uma vez que ela procura posicionar os elementos

combustíveis de modo a manter a média do k-infinito local o mais próximo possível da média do k-infinito global.

A Tabela 6.5 nos mostra os resultados da heurística do K Infinito Médio com a função aptidão para a maximização da concentração de boro.

	Número de Ilhas	**Geração**	**Pot. Média**	**Boro**
Caso 1	1	151	1,356	1138
Caso 2	1	180	1,406	955
Caso 1	3	251	1,387	1174
Caso 2	3	155	1,364	1368
Caso 1	5	180	1,367	1138
Caso 2	5	95	1,382	1141
Caso 1	7	206	1,386	1174
Caso 2	7	141	1,391	1054
	Valores médios	**169**	**1,380**	**1142**

Tabela 6.5 – Resultados da heurística K Infinito Médio para maximização de boro

A Tabela 6.6 nos mostra os resultados da heurística do K Infinito Médio com a função aptidão para a minimização da potência média.

	Número de Ilhas	**Geração**	**Pot. Média**	**Boro**
Caso 1	1	180	1,416	955
Caso 2	1	57	1,375	1150
Caso 1	3	126	1,313	961
Caso 2	3	103	1,357	1086
Caso 1	5	204	1,306	1086
Caso 2	5	125	1,354	1096
Caso 1	7	197	1,345	931
Caso 2	7	261	1,363	1149
	Valores médios	**156**	**1,354**	**1052**

Tabela 6.6 – Resultados da heurística K Infinito Médio para minimização da potência média

6.2.4 Heurística das Regiões

A quarta heurística local divide o oitavo de núcleo em três regiões, região exterior, região mediana e região central, como mostra a Figura 5.4 e faz a distribuição dos elementos combustíveis da seguinte forma: Nas regiões exterior e mediana são colocados elementos combustíveis com k-infinito próximos e na região central é feita uma distribuição do tipo tabuleiro de xadrez, ou seja, com elementos de k-infinito alto e baixo alternadamente.

A Tabela 6.7 nos mostra os resultados da heurística das Regiões com a função aptidão para a maximização da concentração de boro.

	Número de Ilhas	**Geração**	**Pot. Média**	**Boro**
Caso 1	1	243	1,434	1011
Caso 2	1	205	1,360	971
Caso 1	3	82	1,415	983
Caso 2	3	10	1,381	1113
Caso 1	5	112	1,390	1000
Caso 2	5	297	1,356	1248
Caso 1	7	13	1,428	1015
Caso 2	7	140	1,393	1144
	Valores médios	**137**	**1,395**	**1060**

Tabela 6.7 – Resultados da heurística das Regiões para maximização de boro

A Tabela 6.8 nos mostra os resultados da heurística das Regiões com a função aptidão para a minimização da potência média.

	Número de Ilhas	**Geração**	**Pot. Média**	**Boro**
Caso 1	1	205	1,360	971
Caso 2	1	282	1,367	950
Caso 1	3	14	1,375	952
Caso 2	3	238	1,362	1066
Caso 1	5	281	1,376	901
Caso 2	5	137	1,368	946
Caso 1	7	144	1,406	1022
Caso 2	7	219	1,342	1089
	Valores médios	**190**	**1,370**	**987**

Tabela 6.8 – Resultados da heurística das Regiões para minimização da potência média

6.2.5 Heurística Baixa Fuga

A quinta heurística local divide, novamente, o oitavo do núcleo em três regiões, região exterior, região mediana e região central, como mostra a Figura 5.4 e faz a distribuição dos elementos combustíveis como se segue: Na região exterior são colocados elementos combustíveis com k-infinito da seguinte forma: se a média, dos k-infinito, dos elementos combustíveis (r,s) for menor do que a média global de todos os elementos combustíveis a heurística é: valor absoluto de (k_inf(r) - k_inf(s))

multiplicado por um fator 2, senão multiplicado por 1, pois, com isso, queremos aumentar o valor da heurística nestas posições, pois estamos em posição exterior e o k-infinto "resultante" destes dois elementos combustíveis é um valor "baixo", com isso tentamos colocar elementos com k-infinito baixos nesta região. Nas regiões mediana e central são colocados elementos combustíveis com k-infinito da seguinte forma: se a média, dos k-infinito, dos elementos combustíveis (r,s) for maior do que a média global de todos os elementos combustíveis a heurística é: valor absoluto de (k_inf(r) - k_inf(s)) multiplicado por um fator 2, senão multiplicado por 1, pois, com isso, queremos aumentar o valor da heurística nestas posições, pois estamos em posição de meio e centro de núcleo e o k-infinto "resultante" destes dois elementos combustíveis é um valor "alto", com isso tentamos colocar elementos com k-infinito altos nesta região.

A Tabela 6.9 nos mostra os resultados da heurística Baixa Fuga com a função aptidão para a maximização da concentração de boro.

	Número de Ilhas	**Geração**	**Pot. Média**	**Boro**
Caso 1	1	233	1,389	996
Caso 2	1	113	1,384	1074
Caso 1	3	165	1,393	1163
Caso 2	3	60	1,390	970
Caso 1	5	119	1,380	1105
Caso 2	5	36	1,372	1233
Caso 1	7	42	1,395	1166
Caso 2	7	122	1,382	1127
	Valores médios	**111**	**1,386**	**1104**

Tabela 6.9 – Resultados da heurística Baixa Fuga para maximização de boro

A Tabela 6.10 nos mostra os resultados da heurística Baixa Fuga com a função aptidão para a minimização da potência média.

	Número de Ilhas	**Geração**	**Pot. Média**	**Boro**
Caso 1	1	108	1,374	984
Caso 2	1	107	1,347	1233
Caso 1	3	53	1,351	886
Caso 2	3	150	1,336	1177
Caso 1	5	276	1,352	1312
Caso 2	5	188	1,393	1016
Caso 1	7	110	1,356	1070
Caso 2	7	28	1,337	1057
	Valores médios	**127**	**1,356**	**1091**

Tabela 6.10 – Resultados da heurística Baixa Fuga para minimização da potência média

6.2.6 Heurística Parasita

A sexta heurística local, que na verdade é uma modificação do algoritmo, "aproveita-se" do aprendizado das formigas, obtido pela atualização do feromônio, e usa este como heurística local da seguinte forma: O algoritmo é iniciado normalmente, com o parâmetro beta com valor zero, depois de um determinado número de gerações, que neste caso em particular foi usado o valor 50, o algoritmo pára sua execução, copia a matriz feromônio integralmente para a matriz heurística, o valor de beta é trocado para

um valor positivo, que neste caso em particular foi usado o valor 1, e preenche a matriz feromônio integralmente com FEzero, ou seja, a matriz feromônio volta a ter seus valores iniciais.

A Tabela 6.11 nos mostra os resultados da heurística Parasita com a função aptidão para a maximização da concentração de boro.

	Número de Ilhas	**Geração**	**Pot. Média**	**Boro**
Caso 1	1	50	1,351	1268
Caso 2	1	56	1,391	1056
Caso 1	3	38	1,384	1269
Caso 2	3	217	1,361	1193
Caso 1	5	130	1,324	1260
Caso 2	5	72	1,393	1093
Caso 1	7	182	1,381	1238
Caso 2	7	203	1,389	1201
	Valores médios	**118**	**1,372**	**1197**

Tabela 6.11 – Resultados da heurística Parasita para maximização de boro

A Tabela 6.12 nos mostra os resultados da heurística Parasita com a função aptidão para a minimização da potência média.

	Número de Ilhas	**Geração**	**Pot. Média**	**Boro**
Caso 1	1	88	1,356	1023
Caso 2	1	73	1,353	978
Caso 1	3	14	1,278	966
Caso 2	3	272	1,314	938
Caso 1	5	24	1,283	967
Caso 2	5	77	1,278	996
Caso 1	7	49	1,315	1041
Caso 2	7	10	1,275	1025
	Valores médios	**75**	**1,306**	**991**

Tabela 6.12 – Resultados da heurística Parasita para minimização da potência média

6.2.7 Heurística Global

A sétima heurística testada foi a heurística Global, ou seja, sem se utilizar nenhuma heurística local. Este teste foi executado com o parâmetro beta, que define a importância da heurística local, com valor zero. Com isso o algoritmo só se utiliza do conhecimento adquirido ao longo das gerações, através da atualização da matriz feromônio.

A Tabela 6.13 nos mostra os resultados da heurística Global com a função aptidão para a maximização da concentração de boro.

	Número de Ilhas	**Geração**	**Pot. Média**	**Boro**
Caso 1	1	50	1,351	1269
Caso 2	1	175	1,392	1105
Caso 1	3	38	1,384	1289
Caso 2	3	62	1,393	1128
Caso 1	5	115	1,391	1206
Caso 2	5	219	1,394	1179
Caso 1	7	220	1,370	1224
Caso 2	7	17	1,346	1166
	Valores médios	**112**	**1,378**	**1195**

Tabela 6.13 – Resultados da heurística Global para maximização de boro

A Tabela 6.14 nos mostra os resultados da heurística Global com a função aptidão para a minimização da potência média.

	Número de Ilhas	**Geração**	**Pot. Média**	**Boro**
Caso 1	1	1	1,384	1039
Caso 2	1	69	1,368	1089
Caso 1	3	14	1,278	966
Caso 2	3	59	1,278	996
Caso 1	5	24	1,283	967
Caso 2	5	42	1,318	849
Caso 1	7	58	1,264	995
Caso 2	7	10	1,275	1025
	Valores médios	**34**	**1,306**	**990**

Tabela 6.14 – Resultados da heurística Global para minimização da potência média

6.2.8 Heurística Global Completa

A oitava heurística testada foi a heurística Global Completa, ou seja, se considerando o problema de forma completa, com cada uma das 20 formigas tendo como elemento combustível inicial cada um dos 20 elementos combustíveis possíveis, totalizando 200 formigas, sem se utilizar nenhuma heurística local. Com isso o algoritmo só se utiliza do conhecimento adquirido ao longo das gerações através da atualização da matriz feromônio.

A Tabela 6.15 nos mostra os resultados da heurística Global Completa com a função aptidão para a maximização da concentração de boro.

	Número de Ilhas	**Geração**	**Pot. Média**	**Boro**
Caso 1	1	14	1,372	1297
Caso 2	1	9	1,372	1297
Caso 1	3	177	1,389	1331
Caso 2	3	55	1,380	1340
Caso 1	5	120	1,380	1340
Caso 2	5	149	1,380	1340
Caso 1	7	103	1,389	1422
Caso 2	7	227	1,389	1422
	Valores médios	**106**	**1,381**	**1348**

Tabela 6.15 – Resultados da heurística Global Completa para maximização de boro

A Tabela 6.16 nos mostra os resultados da heurística Global Completa com a função aptidão para a minimização da potência média.

	Número de Ilhas	**Geração**	**Pot. Média**	**Boro**
Caso 1	1	8	1,264	995
Caso 2	1	8	1,264	995
Caso 1	3	197	1,244	1026
Caso 2	3	9	1,264	995
Caso 1	5	31	1,243	994
Caso 2	5	26	1,243	995
Caso 1	7	68	1,243	994
Caso 2	7	19	1,243	995
	Valores médios	**45**	**1,251**	**998**

Tabela 6.16 – Resultados da heurística Global Completa para minimização da potência média

6.2.9 Heurística do Nivelamento Completa

A nona heurística testada foi a heurística do Nivelamento Completa, ou seja, considerando-se o problema de forma completa, com cada uma das 20 formigas tendo como elemento combustível inicial cada um dos 20 elementos combustíveis possíveis, totalizando 200 formigas, utilizando-se a heurística do Nivelamento definida na seção 6.2.1.

A Tabela 6.17 nos mostra os resultados da heurística do Nivelamento Completa com a função aptidão para a maximização da concentração de boro.

	Número de Ilhas	**Geração**	**Pot. Média**	**Boro**
Caso 1	1	137	1,366	1239
Caso 2	1	167	1,371	1153
Caso 1	3	56	1,371	1153
Caso 2	3	66	1,371	1153
Caso 1	5	47	1,389	1421
Caso 2	5	46	1,395	1232
Caso 1	7	25	1,373	1347
Caso 2	7	183	1,378	1389
	Valores médios	**91**	**1,377**	**1261**

Tabela 6.17 – Resultados da heurística do Nivelamento Completa para maximização de boro

A Tabela 6.18 nos mostra os resultados da heurística do Nivelamento Completa com a função aptidão para a minimização da potência média.

	Número de Ilhas	**Geração**	**Pot. Média**	**Boro**
Caso 1	1	66	1,298	1053
Caso 2	1	16	1,298	1053
Caso 1	3	133	1,295	1052
Caso 2	3	166	1,294	1052
Caso 1	5	81	1,294	1052
Caso 2	5	77	1,294	1052
Caso 1	7	257	1,287	1142
Caso 2	7	161	1,294	1052
	Valores médios	**120**	**1,294**	**1064**

Tabela 6.18 – Resultados da heurística do Nivelamento Completa para minimização da potência média

6.2.10 Heurística Parasita Completa

A décima heurística testada foi a heurística Parasita Completa, ou seja, se considerando o problema de forma completa, com cada uma das 20 formigas tendo como elemento combustível inicial cada um dos 20 elementos combustíveis possíveis, totalizando 200 formigas, utilizando-se a heurística do parasita definida na seção 6.2.6.

A Tabela 6.19 nos mostra os resultados da heurística Parasita Completa com a função aptidão para a maximização da concentração de boro.

	Número de Ilhas	**Geração**	**Pot. Média**	**Boro**
Caso 1	1	14	1,372	1297
Caso 2	1	9	1,372	1297
Caso 1	3	9	1,371	1297
Caso 2	3	35	1,380	1339
Caso 1	5	184	1,392	1368
Caso 2	5	141	1,380	1340
Caso 1	7	235	1,386	1424
Caso 2	7	50	1,368	1372
	Valores médios	**85**	**1,378**	**1342**

Tabela 6.19 – Resultados da heurística Parasita Completa para maximização de boro

A Tabela 6.20 nos mostra os resultados da heurística Parasita Completa com a função aptidão para a minimização da potência média.

	Número de Ilhas	**Geração**	**Pot. Média**	**Boro**
Caso 1	1	8	1,264	995
Caso 2	1	8	1,264	995
Caso 1	3	268	1,243	995
Caso 2	3	76	1,255	1023
Caso 1	5	31	1,243	994
Caso 2	5	26	1,243	995
Caso 1	7	51	1,243	995
Caso 2	7	248	1,244	1026
	Valores médios	**90**	**1,250**	**1002**

Tabela 6.20 – Resultados da heurística Parasita Completa para minimização da potência média

6.3 Resultados Comparativos das Heurísticas

Uma boa heurística é aquela que acha melhores resultados em menores tempos. Para o problema da recarga, isto é extremamente difícil de se conseguir, pois com esta heurística "mágica" nas mãos do especialista o problema se tornaria de fácil solução, pois seria apenas uma questão de aplicá-la. A heurística do especialista é o conhecimento e a experiência. O algoritmo RCCA tem uma heurística global, que funciona de forma similar a outros algoritmos evolucionários, mas o diferencial deste

algoritmo é a sua heurística local, que se traduz na capacidade de ao se começar a busca, essa busca inicial não ser aleatória, como acontece com outros algoritmos, mas sim com uma direção coerente a ser seguida. A dificuldade é conseguir uma heurística que nos diga esta direção de uma forma mais correta possível, pois para um problema de extrema dificuldade, com grandezas não lineares e multimodal, isto se torna uma tarefa difícil, senão impossível.

As heurísticas utilizadas neste trabalho foram uma tentativa de se conseguir esta "direção coerente" na busca inicial e agora apresentaremos os gráficos comparativos das heurísticas utilizadas.

Os resultados da função cujo objetivo era a maximização da concentração de boro estão a seguir.

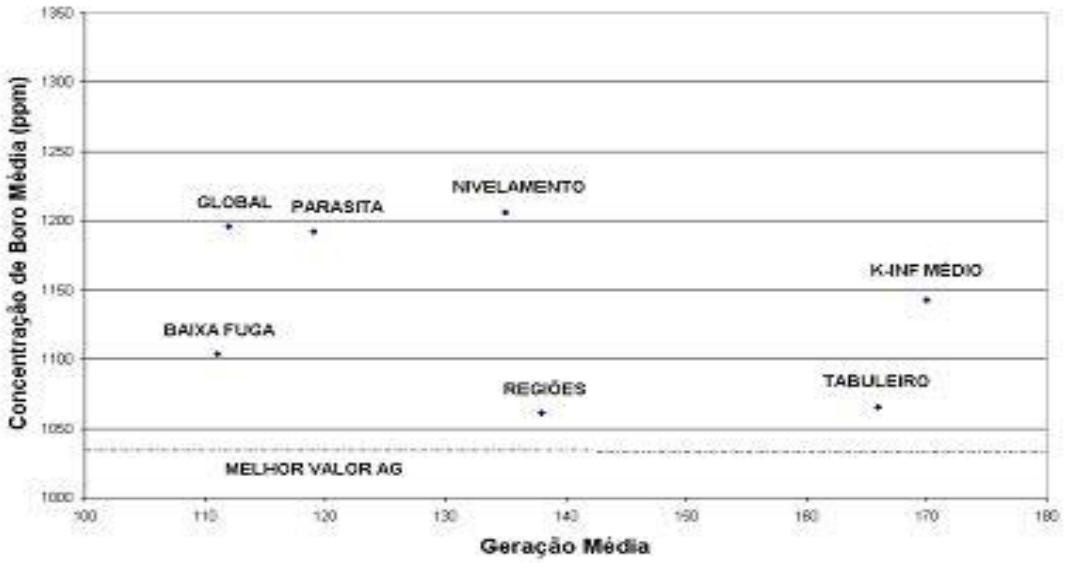

Gráfico 6.1 – Número de gerações médio versus Concentração de boro média – casos simplificados

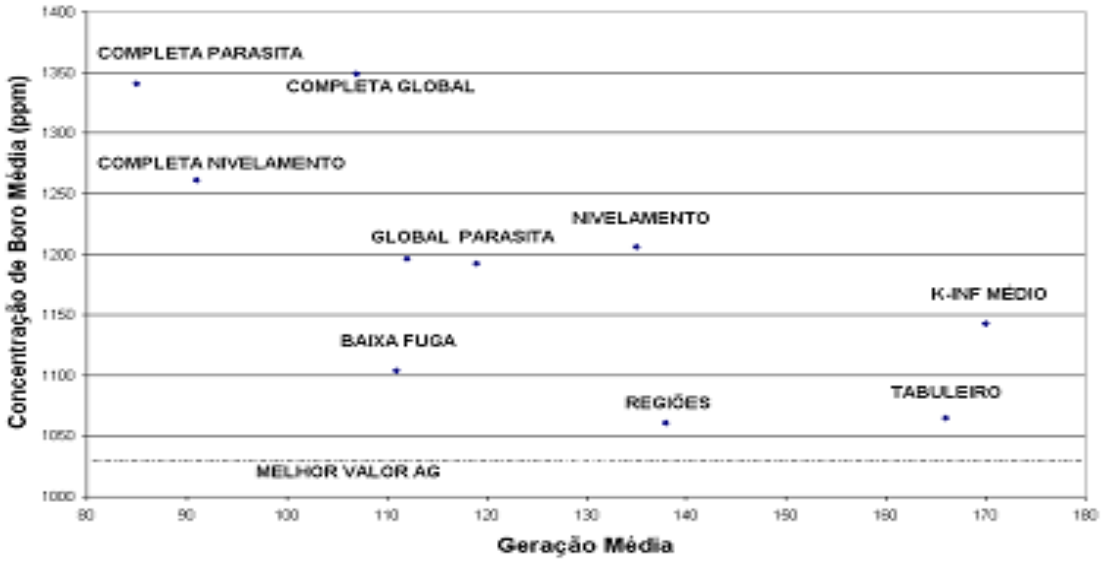

Gráfico 6.2 – Número de gerações médio versus Concentração de boro média – todos os casos

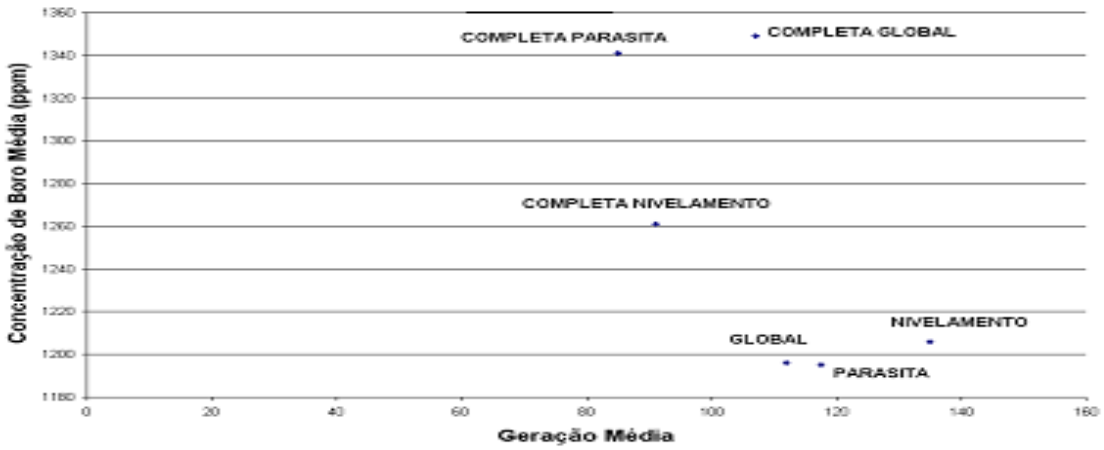

Gráfico 6.3 – Número de gerações médio versus Concentração de boro média – melhores resultados

Nos Gráficos 6.1, 6.2 e 6.3 podemos observar o desempenho de cada heurística em relação às outras. Lembrando que quanto mais o ponto se aproximar do canto superior esquerdo, melhor foi o desempenho da heurística. Podemos verificar, no Gráfico 6.1, onde só temos os casos simplificados, que o ponto ideal seria o que mais se aproximasse do vértice superior esquerdo, pois neste ponto a concentração de boro seria máxima e o número de gerações, para se atingir este ponto, seria mínimo. Notamos que as três melhores heurísticas são a do Nivelamento, a Global e a Parasita, pois são as três que mais se aproximam da situação ideal. Mas, neste gráfico, temos que considerar a concentração de boro média uma grandeza mais importante do que a geração média, pois todos os algoritmos convergiram dentro de um número aceitável de gerações, então as melhores heurísticas são as que têm maiores concentrações de boro, mesmo assim as três heurísticas obtiveram os melhores desempenhos.

No Gráfico 6.2, temos todos os casos estudados em um mesmo gráfico, os simplificados e os completos. Podemos perceber que os resultados dos casos completos são bem superiores aos simplificados, estes últimos que em uma situação de emergência e com uma boa heurística nos forneceria resultados aceitáveis em um tempo bem menor, pois seus resultados são bem superiores aos do AG.

E no Gráfico 6.3, só temos os três melhores casos simplificados e seus três correspondentes completos. Podemos notar que para cada caso simplificado o seu resultado correspondente ao completo é bem superior. Vale ressaltar que a heurística denominada Completa é a heurística Global com uma formiga saindo de cada elemento combustível e em cada posição do núcleo, ou seja, 200 formigas, 10 vezes mais do que as outras heurísticas. Todas as heurísticas, até as que obtiveram os piores resultados,

obtiveram valores médios superiores ao melhor valor encontrado pelo algoritmo genético (CHAPOT, 2000).

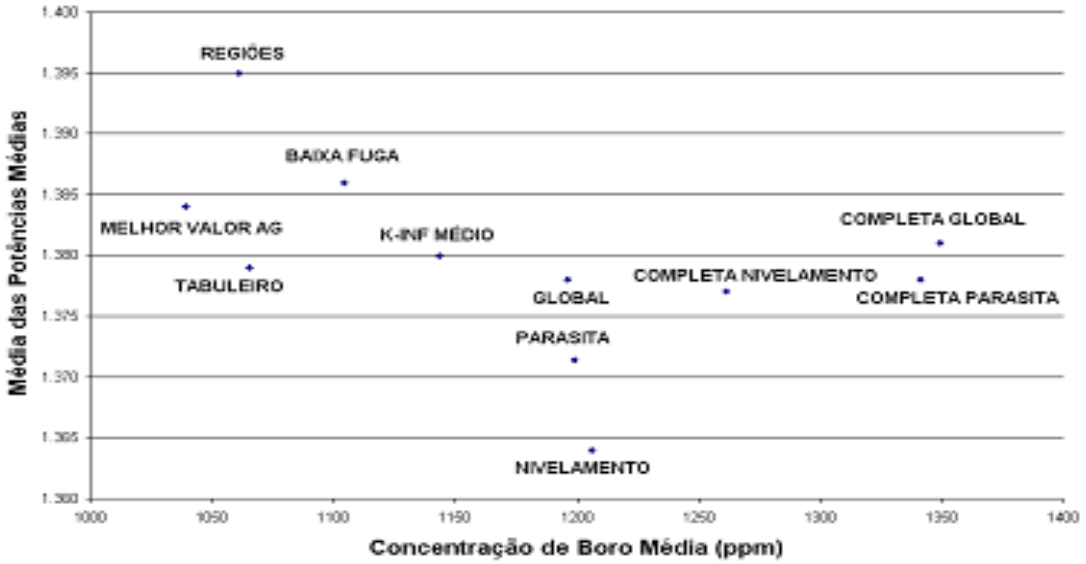

Gráfico 6.4 – Concentração de boro média versus Média das Potências Médias

No Gráfico 6.4 podemos observar que o ponto ideal seria o que mais se aproximasse do vértice inferior direito, pois neste ponto está a maior concentração de boro e a menor média das potências médias e mais uma vez a heurística Completa obteve o melhor resultado. Levando-se em consideração de que o limite da potência média utilizado foi de 1,395 e que todos os resultados estão abaixo deste, os melhores valores são os que estão mais à direita, assim mesmo os melhores resultados foram gerados pelas três heurísticas Completas, com a maior concentração de boro média obtida.

A seguir estão os resultados da função cujo objetivo é a minimização da potência média.

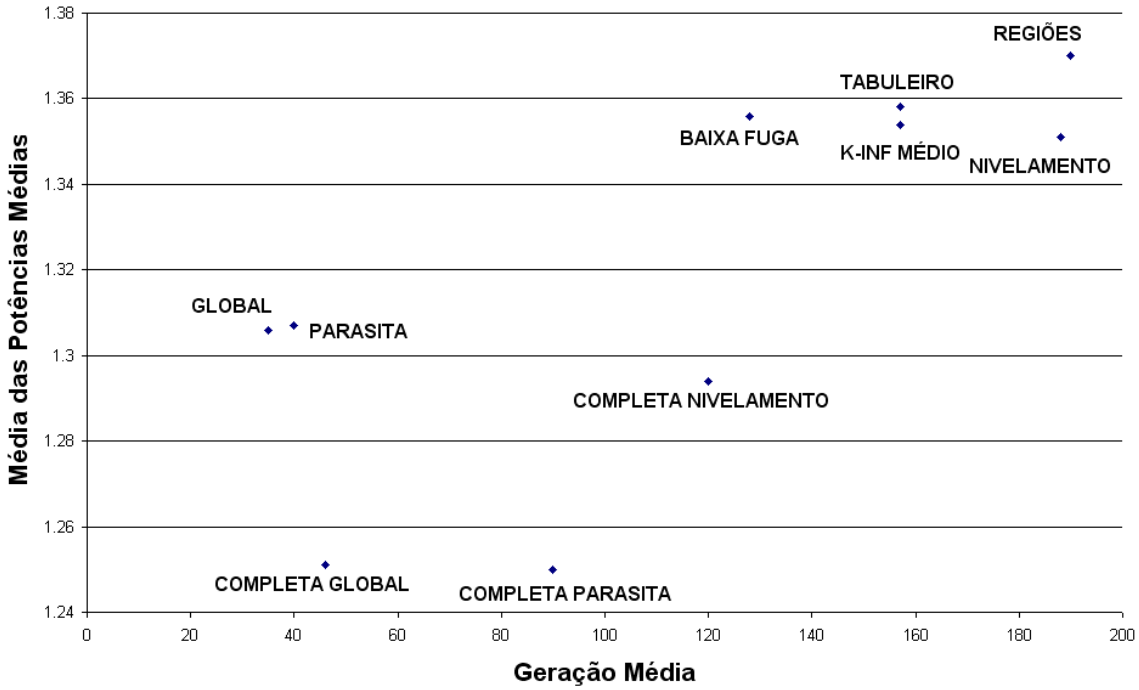

Gráfico 6.5 – Geração média versus Média das Potências Médias

No Gráfico 6.5 podemos notar que o ponto ideal seria o que mais se aproximasse do vértice inferior esquerdo, pois neste ponto teríamos a menor potência média com um número mínimo de gerações, neste caso as heurísticas Global, Parasita e Completas obtiveram os melhores resultados, mas levando em consideração que todos os algoritmos conseguiram obter resultados abaixo do valor limite de 1,395 e que todos conseguiram convergir em um número aceitável de gerações, devemos indicar como melhor resultado encontrado o de menor valor de potência média, e o melhor valor encontrado foi o da heurística Completa Parasita, com 1,250 de Média das Potências Médias.

Os melhores resultados absolutos de todas as heurísticas são apresentados nas Tabelas 6.21 e 6.22.

Heurística	**Número de Ilhas**	**Geração**	**Pot. Média**	**Boro**
Nivelamento	5	93	1,384	1360
Tabuleiro de xadrez	7	11	1,393	1176
K Inf Médio	3	155	1,364	1368
Regiões	5	297	1,356	1248
Baixa Fuga	5	36	1,372	1233
Parasita	3	38	1,384	1269
Global	3	38	1,384	1289
Completa Global	7	103	1,389	1422
Completa Nivelamento	5	47	1,389	1421
Completa Parasita	7	235	1,386	1424
AG	**-**	**-**	**1,390**	**1026**

Tabela 6.21 – Melhores resultados absolutos para a função objetivo de maximização da concentração de boro

Heurística	Número de Ilhas	Geração	Pot. Média	Boro
Nivelamento	3	296	1,308	895
Tabuleiro de xadrez	3	196	1,320	993
K Inf Médio	5	204	1,306	1086
Regiões	7	219	1,342	1089
Baixa Fuga	7	219	1,342	1089
Parasita	7	10	1,275	1025
Global	7	58	1,264	995
Completa Global	7	19	1,243	995
Completa Nivelamento	5	77	1,294	1052
Completa Parasita	5	26	1,243	995

Tabela 6.22 – Melhores resultados absolutos para a função objetivo de minimização da potência média

Na Tabela 6.23 temos os valores médios de todas as heurísticas, para a função maximização da concentração de boro, e o melhor valor encontrado pelo AG.

Heurística	**Geração**	**Pot. Média**	**Boro**
Nivelamento	134	1,364	1206
Tabuleiro de xadrez	165	1,379	1064
K Inf Médio	169	1,380	1142
Regiões	137	1,395	1060
Baixa Fuga	111	1,386	1104
Parasita	118	1,372	1197
Global	112	1,378	1195
Completa Global	106	1.381	1348
Completa Nivelamento	91	1,377	1261
Completa Parasita	85	1,378	1342
AG	**-**	**1,390**	**1026**

Tabela 6.23 – Resultados médios para a função objetivo de maximização da concentração de boro

Na Tabela 6.23 podemos observar que todos os valores médios são melhores do que o melhor valor encontrado pelo AG, levando em consideração apenas a concentração de boro.

CAPÍTULO 7

7. Conclusão e Propostas para Trabalhos Futuros

Neste capítulo são apresentadas as conclusões e as propostas para trabalhos futuros utilizando o Sistema Redes Conectivas de Colônias Artificiais.

7.1 Conclusão

Podemos perceber pelos resultados apresentados no Capítulo 6 que o modelo de paralelização denominado Modelo de Ilhas aplicado ao algoritmo Sistema de colônias de Formigas (SCF), que foi denominado Redes Conectivas de Colônias Artificiais (RCCA), melhorou os resultados do SCF serial (1 ilha) em todos os casos analisados e em todos os casos foi superior aos resultados do AG.

O número de execuções do algoritmo foi de 160 vezes, com 16 vezes para cada uma das 6 heurísticas locais testadas, 16 vezes para a heurística global e mais 48 vezes para as três heurísticas Completas.

Quanto ao número de gerações para convergência do algoritmo, todas as execuções tiveram o número máximo de gerações igual a 300, e observamos que o número mínimo médio de gerações para a convergência do algoritmo foi na heurística Global e igual a 34, com a função objetivo para a minimização do fator de pico e o

número máximo médio de gerações foi na heurística Regiões e igual a 190, com a função objetivo para a minimização do fator de pico.

Os melhores resultados médios, para a função objetivo maximização da concentração de boro, foram obtidos pelas heurísticas Completas, por sua consistência em achar resultados muito bons em todas as execuções. A heurística Completa Global com o valor médio de 1348 ppm de boro, a Completa Parasita com 1342 ppm de boro foram os dois melhores resultados e o melhor valor dos casos simplificados foi o da heurística Nivelamento com 1206 ppm de boro. Vale ressaltar que algumas concentrações de boro encontradas seriam altas demais e talvez não servissem para a recarga, devido a limitações da Especificação Técnica de Angra 1, o que invalidaria tal arranjo de núcleo. No entanto, lembra-se que na função objetivo utilizada não foi dada nenhuma penalização para tal acontecimento.

Os melhores resultados, para a função objetivo minimização da potência média, para valores absolutos, foram obtidos pelas heurísticas Completa Global e Completa Parasita, sendo igual a 1,243 e para valor médio a melhor heurística foi a Completa Parasita com 1,250.

Este trabalho utilizou uma variável de posição P, que nos fornece uma melhor visão de vizinhança, nos dizendo quem são os EC vizinhos e em que posição do núcleo se encontram, pois o EC *x* ao lado do EC *y* na periferia do núcleo contribui de uma forma diferente do EC *x* ao lado do EC *y* no meio do núcleo.

Este trabalho também mostrou que fazer uma analogia direta do PCV com o problema da recarga, comparando-se as cidades com os EC e as rotas com as configurações de núcleo, não é a forma mais correta, pois no PCV a cidade *a* vizinha da cidade *b* contribui da mesma forma para a solução final, independentemente da posição da rota em que se encontram. O mesmo não é verdade no problema da recarga. Por exemplo, um PCV de 20 cidades teria uma configuração de 20 formigas, cada uma saindo de cada cidade. Já um problema de recarga (sem distinção entre quartetos e octetos) com 20 EC, teria uma configuração de 400 formigas, cada qual saindo de um EC, com cada EC ocupando as 20 posições possíveis de núcleo, totalizando um total de 400 formigas.

Lembramos que esses resultados foram obtidos variando-se apenas poucos parâmetros básicos, apenas 2 dos 9 existentes, pois todos os testes foram realizados em uma máquina, que simulava execuções em paralelo. Com isso, podemos sugerir que o Modelo de Ilhas aplicado ao algoritmo SCF é uma técnica de paralelização viável para se obter melhorias em resultados de problemas combinatórios complexos.

Este trabalho nos fornece uma nova ferramenta para o problema da recarga do reator nuclear tipo *PWR*, pois com os resultados alcançados obtivemos uma melhora nos resultados do AG e com isso podemos ter uma expectativa de ganho na recarga real, pois, mesmo o RECNOD sendo um código de física de reatores que não possui alguns módulos importantes, como por exemplo a reconstrução pino a pino, seus resultados são comparáveis a qualquer código existente no mercado.

7.2 – Propostas futuras

- Alterar todos os parâmetros restantes do algoritmo RCCA para uma tentativa de se obter uma melhora nos resultados atuais, pois alguns destes parâmetros são muito sensíveis ao problema e não foram alterados para se manter uma coerência na análise dos dados.

- Utilizar diferentes parâmetros e/ou diferentes heurísticas e/ou diferentes funções objetivo em cada ilha, sendo equivalente a cada ilha ser um formigueiro de formigas de diferentes espécies.

- Acoplar o RCCA a um código comercial de física de reatores, para se tentar fazer a recarga real de Angra 1.

- Pesquisar outras heurísticas locais, para uma tentativa de o algoritmo convergir mais rapidamente com valores aceitáveis, pois um dos grandes problemas da recarga nuclear é o tempo de execução dos códigos de física de reatores, que é extremamente alto.

- Conjugar a heurística Completa, que nos forneceu os melhores resultados, com outras heurísticas locais.

- Utilizar uma rede de computadores interligados entre si e acoplados a códigos de física de reatores, com a finalidade de se realizar uma busca em paralelo real de boas soluções em tempos mais razoáveis.

APÊNDICE A

A. Uma breve introdução sobre Agentes Artificiais

Os agentes artificiais são alvo de interesse em pesquisa em diversas áreas de conhecimento, não ficando mais restritos, como há alguns anos atrás, à pesquisa em Inteligência Artificial. Entre as áreas de conhecimento que também estudam e utilizam os agentes artificiais destacam-se: a psicologia, a sociologia, a comunicação e a educação.

Esse interesse interdisciplinar, despertado pelos agentes artificiais, fez com que surgissem várias tentativas de definições do termo agente artificial e das suas funcionalidades. Devido a este fato, tornou-se difícil a obtenção de um consenso entre os pesquisadores a respeito de uma única definição para o que seria exatamente um agente artificial e de como ele trabalharia.

Na tentativa de explicação da afirmação acima são enumeradas algumas das definições de agentes encontradas na literatura:

- Agentes de *software* são programas de computadores que aplicam técnicas de Inteligência Artificial para prover assistência ao usuário em tarefas computacionais. (MAES, 1995).

- A palavra agente deve ser usada para referenciar uma máquina que executa algo sem que seja necessário sabermos como ela trabalha, algo semelhante a uma "caixa preta". (MINSK, 1994).

- Agentes são entidades quaisquer que percebem o ambiente em que se encontram através de sensores e agem sobre este mesmo ambiente através de atuadores. (RUSSEL e NORVIG, 1995).

- Um agente é um *hardware* ou (mais frequentemente) um *software* que desfruta das seguintes propriedades:

    1. Autonomia

       Os agentes operam sem intervenção humana e devem ter uma espécie de controle sobre suas ações e estados internos.

    2. Cooperação

       Os agentes interagem com outros agentes (e possivelmente com seres humanos) através de alguma linguagem específica.

    3. Reatividade

       Os agentes percebem seu ambiente (que pode ser o mundo físico, uma interface de usuários, uma coleção de outros agentes, a Internet ou até todos esses ambientes combinados) e respondem às alterações que ocorram neste ambiente.

    4. Pró-atividade

> Os agentes não só respondem às alterações ambientes como também estão aptos a exibirem um comportamento direcionado aos objetivos tomando iniciativas (WOOLDRIDGE e JENNINGS, 1995).

As definições acima demonstram claramente a falta de uma definição formal para o termo agente. É fácil notar que elas foram crescendo em grau de complexidade à medida que são enumeradas, porém entre essas definições pode-se encontrar uma linha mestra que condense, de modo simplificado, um conceito básico de agentes.

É nesse conceito básico, que envolve apenas os pontos de similaridade existentes entre todos os agentes e seu funcionamento primário, que se pretende concentrar a próxima seção, deixando as particularidades de cada tipo de agente para uma futura classificação.

A.1. Definição básica de Agentes Autônomos

O termo agente foi inicialmente utilizado em trabalhos de Inteligência Artificial nos quais os pesquisadores procuravam criar uma entidade artificial que reproduzisse as habilidades humanas. Porém, devido ao uso indiscriminado deste termos pelas indústrias de *software*, as quais utilizaram-no de diferentes formas e com vários significados como ferramenta de apelo popular para a aceitação de seus produtos, tornou-se difícil uma definição clara, precisa e única para os agentes artificiais. Desta

forma, esta seção propõe-se a expor as características principais que todos os agentes devem possuir deixando suas particularidades para uma classificação posterior.

A essência da definição dos agentes encontra-se em um, a saber: **agentes são entidades que executam tarefas.** Porém esse conceito básico não define exatamente um agente. Algumas características essenciais devem fazer parte do conceito principal dos agentes. Essas características serão descritas abaixo.

Pode-se dizer que os agentes habitam um ambiente do qual fazem parte. Cada um dos agentes percebe seu ambiente e pode atuar automaticamente sobre ele, não sendo necessária que nenhuma outra entidade lhe propicione dados de entrada ou que interprete seus dados de saída.

Os agentes são capazes de agir no sentido de buscar seus próprios objetivos como também, de buscar os objetivos traçados por outros agentes. Suas ações conseguem alterar o ambiente ao qual pertencem e também podem perceber as alterações que eles mesmos causam no ambiente.

E por último, os agentes executam suas ações continuamente no tempo.

De acordo com as características básicas de um agente citadas acima, pode-se formalizar uma definição de agentes autônomos de seguinte forma:

Um agente autônomo é um sistema que faz parte do ambiente ao qual ele habita; ele é capaz de perceber e de alterar esse ambiente, continuamente no tempo, de

acordo com seus próprios objetivos. A alteração no ambiente causada por um agente é, também, percebida pelo agente.

A definição de agente adotada conduz a uma grande e variada classificação dos agentes, uma vez que dela só faz parte à essência de um agente deixando todas as particularidades de cada tipo para uma classificação posterior.

As classificações dos agentes serão discutidas na próxima seção.

A Figura A.1 ilustra a definição de agentes autônomos descrita acima.

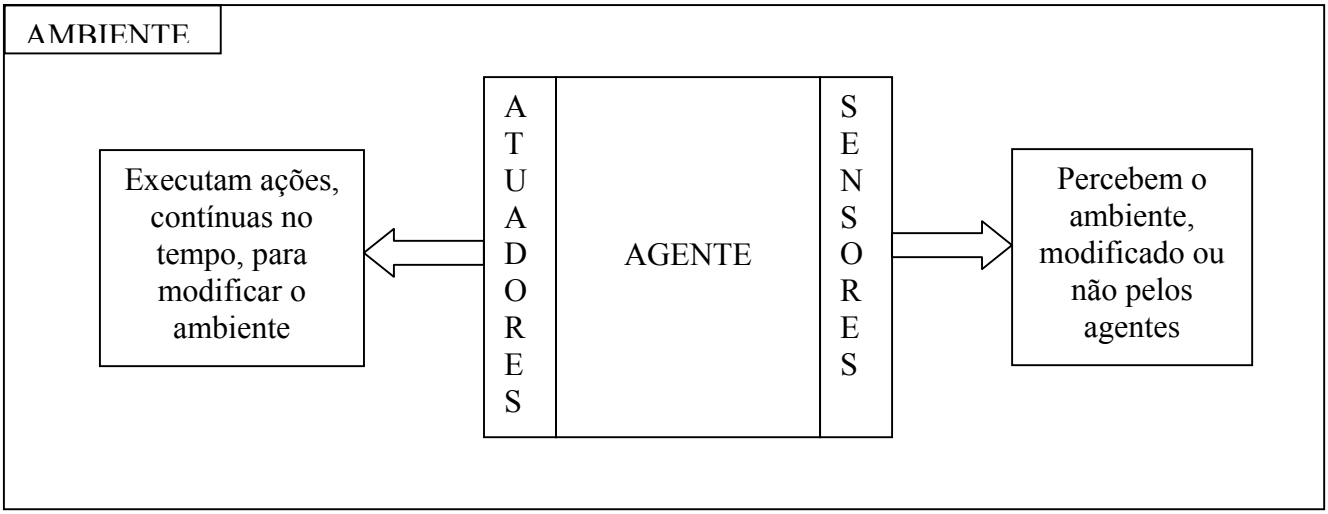

Figura A.1 – Agentes Autônomos

A.2. Classificação dos Agentes

A classificação dos tipos de agentes, assim como sua definição formal, não está padronizada (FRANKLIN e GRAENSER, 1996), (CARDIERI, 1998). Vários pesquisadores classificam os agentes segundo sua funcionalidade (GREEN et al, 1997), enquanto que outros pesquisadores os classificam de acordo com as propriedades que os mesmos apresentam (FRANKLIN e GRAENSER, 1996).

Seguindo a definição de agentes que foi apresentada na seção anterior, na qual apontou-se como agentes as entidades que possuíssem determinadas propriedades, será apresentada inicialmente, e de forma mais detalhada, uma classificação para agentes derivada de suas propriedades.

Antes de iniciar essa classificação serão expostas as principais características encontradas nos vários tipos de agentes existentes. Algumas delas o leitor já tomou conhecimento na seção anterior onde foram apresentadas algumas definições de agentes encontradas na literatura e onde se definiu o termo agente a partir de suas propriedades essenciais.

As principais propriedades encontradas nos diversos tipos de agentes artificiais são:

- Reatividade

  O agente percebe o seu ambiente e responde prontamente as alterações que surjam neste ambiente.

- Autonomia

Os agentes operam sem a intervenção humana possuindo um controle sobre suas próprias ações e seu estado interno.

- Pró-atividade

  Os agentes não respondem apenas às mudanças no seu ambiente, eles também estão aptos a exibir um comportamento visando alcançar metas.

- Continuidade Temporal

  Os agentes são processos que estão em contínua execução, podendo estar ativos ou passivos.

- Cooperação

  Os agentes interagem com outros agentes e possivelmente com seres humanos através de uma linguagem específica.

- Aprendizagem ou Adaptabilidade

  Os agentes devem estar aptos a aprender

- Mobilidade

  É a habilidade que os agentes possuem de moverem-se por uma rede.

- Flexibilidade

  As ações dos agentes não seguem um roteiro rígido, elas podem se modificar de acordo com as alterações do ambiente.

- Racionalidade

  Os agentes sempre agirão de modo a alcançarem seus objetivos e nunca de maneira a prejudicá-los.

A classificação dos agentes segundo suas propriedades é realizada informando o subconjunto ao qual o agente pertence, ou seja, informando quais das propriedades acima definidas o agente possui.

Da definição de agente autônomo, exposta na seção anterior, conclui-se que qualquer agente deve possuir ao menos as quatro primeiras propriedades, que são: reatividade, autonomia, pró-atividade e continuidade temporal. Seguindo as demais propriedades que os agentes possuírem é que novas classes de agentes vão sendo construídas. Assim se, além das quatro primeiras propriedades, o agente possuir a propriedade mobilidade este agente é classificado como um agente móvel; ou se possuir a propriedade de aprendizagem ele pertencerá à classe dos agentes que aprendem. Um agente possuidor de ambas as propriedades – mobilidade e aprendizagem – pertencerá à classe dos agentes móveis que aprendem e assim sucessivamente.

Um outro tipo de classificação de agentes tem por base a funcionalidade e a forma de atuação dos agentes. Uma das formas de se dividir esta classificação pode ser a seguinte: 1) Agentes de Interface, 2) Agentes Colaborativos (arquitetura multi-agentes), 3) Agentes de Informação ou Agentes de Internet, 4) Agentes Móveis.

Os Agentes de Interface são aqueles que auxiliam na operação de uma interface interativa. Os agentes inteligentes de interface deferem-se das interfaces inteligentes de usuários, uma vez que os primeiros são dotados de um determinado grau de autonomia. O comportamento dos agentes de interface é semelhante ao comportamento de um assistente pessoal. O usuário pode desprezar o agente se lhe for conveniente.

Os agentes colaborativos são aqueles que trocam informações entre si. Agentes experientes podem ajudar um novo agente a aprender com maior rapidez. A maior parte desse tipo de agente pertence aos sistemas multi-agentes que são sistemas compostos de agentes individuais (ou em alguns casos de agentes e humanos) que trabalham juntos

para resolverem sistemas complexos, assim o resultado final é baseado no resultado de cada agente individualmente.

Os agentes de informação ou de Internet são aqueles que atuam como ferramenta na busca na Internet e ajudam a administrar a infinidade de informações existentes na rede.

Os agentes móveis são aqueles que se deslocam em redes de computadores. A Mobilidade dos agentes está presente em ambas as classificações descritas, uma vez que é tanto um atributo do agente, quanto a sua própria função.

Existem, ainda, várias outras formas de classificação dos agentes que não serão discutidas neste trabalho, mas que podem ser conhecidas através das referências (FRANKLIN e GRAENSER, 1996), (SILVA, 1998).

A.3. Sistemas de Multi-Agentes e Comunicação entre Agentes

A Inteligência Artificial Distribuída (IAD) faz parte de um sub-campo da Inteligência Artificial onde são pesquisados modelos de conhecimento, técnicas de raciocínio e de comunicação que os agentes utilizarão para participar de *sociedades* que podem ser compostas por computadores e humanos.

Uma das áreas estudadas pela IAD, são os sistemas de multi-agentes. Esses sistemas podem ser pensados como compostos de diversos agentes (que podem ser

heterogêneos entre si) que assumem, cada um, uma parte na solução do problema e por um conjunto de caminhos de comunicação entre eles. A solução final será obtida a partir da cooperação de todos os agentes do sistema.

A comunicação entre os agentes é parte fundamental na solução do problema, uma vez que é através dela que os agentes trocam informações a respeito do estado atual do problema e coordenam suas atividades para poderem com isso buscar as soluções.

Classicamente, as arquiteturas de comunicação entre os agentes enquadram-se em duas classes: 1) Arquitetura de Quadro-Negro e 2) Arquitetura de Transmissão de Mensagens.

Na Arquitetura de Quadro-Negro um conjunto de agentes independentes trabalha de modo cooperativo ao redor de um quadro-negro. Este quadro-negro é basicamente um repositório de dados compartilhados onde os agentes escrevem mensagens, depositam resultados e obtêm informações.

O outro sistema de comunicação entre os agentes é o Sistema de Transmissão de Mensagens onde existe um módulo de raciocínio que envia mensagens aos outros módulos de forma específica. Essas mensagens podem ser solicitações de serviços ou informações sobre os resultados dos serviços executados.

Os sistemas de multi-agentes apresentam vantagens significativas com relação aos sistemas monolíticos. Algumas dessas vantagens são enumeradas a seguir:

1. Devido à modularidade desses sistemas é mais fácil criar e executar a sua manutenção, uma vez que alterar uma coleção de módulos computacionais independentes, ou quase independentes, é muito mais fácil do que alterar um imenso programa computacional.
2. Devido ao paralelismo intrínseco desses sistemas torna-se mais rápido a resolução de problemas complexos, além de que eles estão mais aptos a serem adaptados à computação paralela que os programas tradicionais.
3. Com o sistema de multi-agentes pode-se aumentar a confiabilidade do sistema computacional, uma vez que o problema está distribuído entre vários agentes e uma falha em algum dos agentes não acarreta erro ou parada completa do sistema, pois os outros agentes serão capazes de substituir o agente que falhou.

Por essas vantagens os sistemas baseados em agentes tornam-se cada vez mais atraentes para os pesquisadores de múltiplas áreas, inclusive na área de otimização de problemas combinatoriais, no qual se encontra o problema da recarga nuclear, ponto alvo deste trabalho.

APÊNDICE B

B.1 – Algoritmos Genéticos - Um Breve Histórico

Até meados do século 19, os naturalistas acreditavam que cada espécie havia sido criada separadamente por um ser supremo ou através de geração espontânea. O trabalho do naturalista Carolus Linnaeus sobre a classificação biológica de organismos despertou o interesse pela similaridade entre certas espécies, levando a acreditar na existência de uma certa relação entre elas. Outros trabalhos influenciaram os naturalistas em direção à teoria da seleção natural, tais como os de Jean Baptiste Lamarck, que sugeriu uma teoria evolucionária no "uso e desuso" de órgãos; e de Thomas Robert Malthus, que propôs que fatores ambientais tais como doenças e carência de alimentos, limitavam o crescimento de uma população.

Depois de mais de 20 anos de observações e experimentos, Charles Darwin apresentou em 1858 sua teoria de evolução através de seleção natural, simultaneamente com outro naturalista inglês Alfred Russel Wallace. No ano seguinte, Darwin publica o seu *On the Origin of Species by Means of Natural Selection* com a sua teoria completa, sustentada por muitas evidências colhidas durante suas viagens a bordo do navio *Beagle.*

Este trabalho influenciou muito o futuro não apenas da Biologia, Botânica e Zoologia, mas também teve grande influência sobre o pensamento religioso, filosófico, político e econômico da época. A teoria da evolução e a computação nasceram praticamente na mesma época: Charles Babbage, um dos fundadores da computação

moderna e amigo pessoal de Darwin desenvolveu sua máquina analítica em 1833. Ambos provavelmente estariam surpresos com a ligação entre estas duas áreas no campo da Inteligência Artificial nos dias de hoje.

Por volta de 1900, o trabalho de Gregor Mendel, desenvolvido em 1865, sobre os princípios básicos de herança genética, foi redescoberto pelos cientistas e teve grande influência sobre os futuros trabalhos relacionados à evolução. A moderna teoria da evolução combina a genética e as idéias de Darwin e Wallace sobre a seleção natural, criando o princípio básico de Genética Populacional: a variabilidade entre indivíduos em uma população de organismos que se reproduzem sexualmente é produzida pela mutação e pela recombinação genética.

Este princípio foi desenvolvido durante os anos 30 e 40, por biólogos e matemáticos de importantes centros de pesquisa. Nos anos 50 e 60, muitos biólogos começaram a desenvolver simulações computacionais de sistemas genéticos. Entretanto, foi John Holland quem começou, formalmente, a desenvolver as primeiras pesquisas no tema. Holland foi gradualmente refinando suas idéias e em 1975 publicou o seu livro *Adaptation in Natural and Artificial Systems*, hoje considerado a literatura de base de Algoritmos Genéticos. Desde então, estes algoritmos vêm sendo aplicados com sucesso nos mais diversos problemas de otimização e aprendizado de máquina.

B.2 – Como funcionam os AG

Inicialmente, é gerada uma população formada por um conjunto aleatório de indivíduos que podem ser vistos como possíveis soluções do problema. Durante o processo evolutivo, esta população é avaliada: para cada indivíduo é dada uma nota, ou índice, refletindo sua habilidade de adaptação a determinado ambiente. Uma porcentagem dos mais adaptados é mantida, enquanto os outros são descartados (darwinismo). Os membros mantidos pela seleção podem sofrer modificações em suas características fundamentais através de mutações e cruzamento (*crossover*) ou recombinação genética gerando descendentes para a próxima geração. Este processo, chamado de reprodução, é repetido até que uma solução satisfatória seja encontrada.

Embora possam parecer simplistas do ponto de vista biológico, estes algoritmos são suficientemente complexos para fornecer mecanismos de busca adaptativos e robustos.

B.3 - Características Gerais dos AG

Algoritmos Genéticos são algoritmos de otimização global, baseados nos mecanismos de seleção natural e da genética. Eles empregam uma estratégia de busca paralela e estruturada baseada em probabilidades, que é voltada em direção ao reforço da busca de pontos de "alta aptidão", ou seja, pontos nos quais a função a ser minimizada (ou maximizada) tem valores relativamente baixos (ou altos).

Apesar de utilizar mecanismos aleatórios em todas as gerações, eles não são caminhadas aleatórias não direcionadas, pois exploram informações históricas para encontrarem novos pontos de busca onde são esperados melhores desempenhos. Isto é feito através de processos iterativos, onde cada iteração é chamada de geração.

Durante cada iteração, os princípios de seleção e reprodução são aplicados a uma população de candidatos que pode variar, dependendo da complexidade do problema e dos recursos computacionais disponíveis. Através da seleção, se determina quais indivíduos conseguirão se reproduzir, gerando um número determinado de descendentes para a próxima geração, com uma probabilidade determinada pelo seu índice de aptidão. Em outras palavras, os indivíduos com maior adaptação relativa têm maiores chances de se reproduzir.

O ponto de partida para a utilização de Algoritmos Genéticos, como ferramenta para solução de problemas, é a representação destes problemas de maneira que os Algoritmos Genéticos possam trabalhar adequadamente sobre eles. A maioria das representações são genotípicas, utilizam vetores de tamanho finito em um alfabeto finito.

Tradicionalmente, os indivíduos são representados genotipicamente por vetores binários, onde cada elemento de um vetor denota a presença (1) ou ausência (0) de uma determinada característica: o seu genótipo. Os elementos podem ser combinados formando as características reais do indivíduo, ou o seu fenótipo. Teoricamente, esta representação é independente do problema, pois uma vez encontrada a representação em vetores binários, as operações padrão podem ser utilizadas, facilitando o seu emprego em diferentes classes de problemas.

A utilização de representações em níveis de abstração mais altos tem sido investigada. Como estas representações são mais fenotípicas, facilitariam sua utilização em determinados ambientes, onde essa transformação "fenótipo - genótipo" é muito complexa. Neste caso, precisam ser criados os operadores específicos para utilizar estas representações.

O princípio básico do funcionamento dos AG é que um critério de seleção vai fazer com que, depois de muitas gerações, o conjunto inicial de indivíduos gere indivíduos mais aptos. A maioria dos métodos de seleção é projetada para escolher preferencialmente indivíduos com maiores notas de aptidão, embora não exclusivamente, a fim de manter a diversidade da população. Um método de seleção muito utilizado é o Método da Roleta, Figura B.1, onde indivíduos de uma geração são escolhidos para fazer parte da próxima geração, através de um sorteio de roleta.

Neste método, cada indivíduo da população é representado na roleta proporcionalmente ao seu índice de aptidão. Assim, aos indivíduos com alta aptidão é dada uma porção maior da roleta, enquanto aos de aptidão mais baixa é dada uma porção relativamente menor da roleta. Finalmente, a roleta é girada um determinado número de vezes, dependendo do tamanho da população, e são escolhidos, como indivíduos que participarão da próxima geração, aqueles sorteados na roleta.

Indivíduo S_i	Aptidão $f(S_i)$	Aptidão relativa
S_1 10110	3.00	0.17
S_2 11000	8.00	0.44
S_3 11110	1.00	0.06
S_4 01001	4.00	0.22
S_5 00110	2.00	0.11

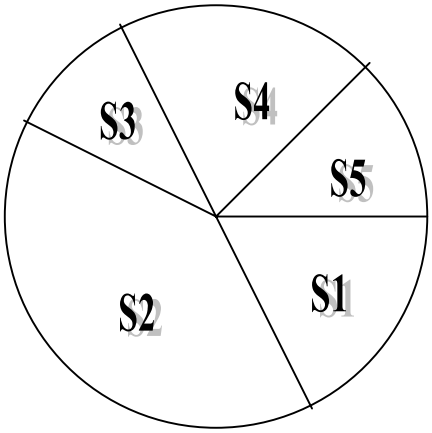

Figura B.1 - Indivíduos de uma população e a sua correspondente roleta de seleção.

Um conjunto de operações é necessário para que, dada uma população, se consiga gerar populações sucessivas que melhorem sua aptidão com o tempo. Estes operadores são: ***cruzamento*** e ***mutação***. Eles são utilizados para assegurar que a nova geração seja totalmente nova, mas ela possui, de alguma forma, características de seus pais, ou seja, a população se diversifica e mantém características de adaptação adquiridas pelas gerações anteriores. Para prevenir que os melhores indivíduos não desapareçam da população pela manipulação dos operadores genéticos, eles podem ser automaticamente colocados na próxima geração, através da reprodução elitista.

Esse ciclo é repetido um determinado número de vezes. Durante esse processo, os melhores indivíduos, assim como alguns dados estatísticos, podem ser coletados e armazenados para avaliação.

Estes algoritmos, apesar de serem computacionalmente muito simples, são bastante poderosos. Além disso, eles não são limitados por suposições sobre o espaço de busca, relativas a continuidade, existência de derivadas, etc.

Buscas em problemas reais são repletas de descontinuidades, ruídos e outros problemas. Métodos que dependam fortemente de restrições de continuidade e existência de derivadas são adequados apenas para problemas em um domínio limitado.

B.4 - Operadores Genéticos

O princípio básico dos operadores genéticos é transformar a população através de sucessivas gerações, estendendo a busca até chegar a um resultado satisfatório. Os operadores genéticos são necessários para que a população se diversifique e mantenha características de adaptação adquiridas pelas gerações anteriores.

O operador de mutação, Figura B.2, é necessário para a introdução e manutenção da diversidade genética da população, alterando arbitrariamente um ou mais componentes de uma estrutura escolhida, como é ilustrado na figura abaixo, fornecendo assim, meios para a introdução de novos elementos na população. Desta forma, a mutação assegura que a probabilidade de se chegar a qualquer ponto do espaço de busca nunca será zero, além de contornar o problema de mínimos locais, pois com este mecanismo, altera-se a direção da busca. O operador de mutação é aplicado aos indivíduos com uma probabilidade dada pela taxa de mutação; geralmente se utiliza uma taxa de mutação pequena, pois é um operador genético secundário.

Antes da Mutação :	1 1 **1** 0 0
Depois da Mutação :	1 1 **0** 0 0

Figura B.2 - Exemplo de mutação.

O cruzamento é o operador responsável pela recombinação de características dos pais durante a reprodução, permitindo que as próximas gerações herdem essas características. Ele é considerado o operador genético predominante, por isso é aplicado com probabilidade dada pela taxa de cruzamento, que deve ser maior que a taxa de mutação.

Este operador pode, ainda, ser utilizado de várias maneiras; as mais utilizadas são:

- um-ponto: um ponto de cruzamento é escolhido e a partir deste ponto as informações genéticas dos pais serão trocadas. As informações anteriores a este ponto em um dos pais são ligadas às informações posteriores à este ponto no outro pai, como é mostrado no exemplo da Figura B.3.

Pai 1 = 00000000000000 | **000000**	Filho 1 = 00000000000000 | **111111**
Pai 2 = 11111111111111 | **111111**	Filho 2 = 11111111111111 | **000000**

Figura B.3 - Um exemplo de cruzamento de um ponto.

- multi-pontos: é uma generalização desta idéia de troca de material genético através de pontos, onde muitos pontos de cruzamento podem ser utilizados, Figura B.4.

Pai 1 = 0000000 | **0000000** | 000000	Filho1 = 0000000 | **1111111** | 000000
Pai 2 = 1111111 | **1111111** | 111111	Filho2 = 1111111 | **0000000** | 111111

Figura B.4 - Um exemplo de cruzamento de multi-pontos.

- uniforme**:** Neste tipo de operador crossover os bits de cada cadeia são trocados alternadamente até o final da cadeia, Figura B.5.

Pai 1 = 00000000000000000000	Filho 1 = 0**1**0**1**0**1**0**1**0**1**0**1**0**1**0**1**
Pai 2 = 11111111111111111111	Filho 2 = 1**0**1**0**1**0**1**0**1**0**1**0**1**0**1**0**

Figura B.5 - Um exemplo de cruzamento uniforme.

B.5 - Parâmetros Genéticos

É importante também, analisar de que maneira alguns parâmetros influem no comportamento dos algoritmos genéticos, para que se possa estabelecê-los conforme as necessidades do problema e dos recursos disponíveis.

Tamanho da População. O tamanho da população afeta o desempenho global e a eficiência dos AG. Com uma população pequena o desempenho pode cair, pois deste modo a população fornece uma pequena cobertura do espaço de busca do problema. Uma grande população geralmente fornece uma cobertura representativa do domínio do

problema, além de prevenir convergências prematuras para soluções locais ao invés de globais. No entanto, para se trabalhar com grandes populações, são necessários maiores recursos computacionais, ou que o algoritmo trabalhe por um período de tempo muito maior.

Taxa de Cruzamento. Quanto maior for esta taxa, mais rapidamente novas estruturas serão introduzidas na população. Mas se esta for muito alta, estruturas com boas aptidões poderão se perder, pois, a maior parte da população será modificada. Com um valor baixo, o algoritmo pode tornar-se muito lento.

Taxa de Mutação. Uma baixa taxa de mutação previne que uma dada posição fique estagnada em um valor, além de possibilitar que se chegue em qualquer ponto do espaço de busca. Com uma taxa muito alta a busca se torna essencialmente aleatória.

Intervalo de Geração. Controla a porcentagem da população que será substituída durante a próxima geração. Com um valor alto, a maior parte da população será substituída, mas com valores muito altos pode ocorrer perda de estruturas de alta aptidão. Com um valor baixo, o algoritmo pode tornar-se muito lento.

B.6 – Uma base matemática dos AG

Uma estrutura importante nos AG são os esquemas. Um esquema é um modelo que descreve um subconjunto da cadeia de bits com similaridade em certas posições desta cadeia. Utilizando um alfabeto ternário {0, 1, *}, formado pela adição do símbolo

* (“não importa”, que indica que a posição pode conter 0 ou 1), vamos considerar, como exemplo, um esquema de 5 bits, H = * 0 0 0 0 que representa duas cadeias de bits, {**1** 0 0 0 0} e {**0** 0 0 0 0} e o esquema H = * 1 1 1 * que representa quatro cadeias de bits, {**0** 1 1 1 **0**}, {**0** 1 1 1 **1**}, {**1** 1 1 1 **0**} e {**1** 1 1 1 **1**}. Desta forma, para uma cadeia de bits de comprimento ***L*** empregando um alfabeto contendo ***k*** caracteres, o número de possíveis esquemas é dado por $(k + 1)^{L}$.

Os esquemas não são criados da mesma forma e alguns trazem mais informações que outros. Para se quantificar estas informações, são apresentadas duas propriedades dos esquemas: a ***ordem do esquema o(H)***, que é o número de posições fixas (0 ou 1) presentes na cadeia de bits, por exemplo, para o esquema H = 0 1 1 * 1 * * o valor de o(H) = 4, e o ***tamanho do esquema δ(H)***, que é a distância entre a primeira e a última posição fixa da cadeia de bits, para o esquema anterior temos δ(H) = 5 – 1 = 4.

Ao longo do processo de busca, os operadores genéticos vão alterando o número de esquemas presentes na população de possíveis soluções.

O efeito do operador de seleção no número de esquemas esperado na próxima população é determinado da seguinte forma: seja ***m*** o número de esquemas ***H*** em um passo ***t***, representado por m = m(H,t). Durante a seleção, as cadeias de bits são escolhidas de acordo com o valor de sua aptidão (*fitness*) segundo uma probabilidade:

$$Pi = \frac{f_i}{\Sigma f_i} \qquad \text{(B.1)}$$

O número de representantes de esquemas ***H*** na população no tempo t + 1, m(H, t+1) é dado pela equação:

$$m(H, t+1) = m(H, t).n.\frac{f(H)}{\sum_j f_j} \tag{B.2}$$

onde: ***n*** = tamanho da população,

f(H) = média da aptidão das cadeias de bits representadas pelo esquema ***H*** no tempo ***t***.

Considerando-se que a média da aptidão de toda a população pode ser dada por:

$$\bar{f} = \sum_{j=1}^{n} \frac{f_i}{n} \tag{B.3}$$

então pode-se reescrever a Equação (B.2) da seguinte forma:

$$m(H,t+1) = m(H,t).\frac{f(H)}{\bar{f}} \tag{B.4}$$

Da Equação (B.4) conclui-se que um esquema cresce com a razão entre a média da aptidão dos esquemas e a média da aptidão da população. Em outras palavras, esquemas com valores de aptidão acima da média da população serão aumentados na próxima geração.

Supondo que um determinado esquema ***H*** se mantenha acima da média da população, com uma quantidade $\bar{c}\,\bar{f}$, com ***c*** constante, pode-se reescrever a Equação (B.4) da seguinte forma:

$$m(H, t+1) = m(H, t).\frac{\bar{f}}{\bar{f}} + c\,\frac{\bar{f}}{\bar{f}} = m(H, t).(1+c) \tag{B.5}$$

Partindo de t = 0 e supondo um valor estacionário para ***c***, tem-se:

$$m(H, t) = m(H, 0).(1+c)^{t} \tag{B.6}$$

O efeito operador de cruzamento sobre a formação dos esquemas parte do princípio que, escolhendo-se aleatoriamente um ponto de corte na cadeia de bits, a probabilidade de um esquema ser destruído é de:

$$P_d = \frac{\delta(H)}{L-1} \tag{B.7}$$

onde: ***L*** é o comprimento da cadeia de bits

e a probabilidade $\boldsymbol{P_s}$, por:

$$P_s = 1 - P_d \Rightarrow P_s = 1 - \frac{\delta(H)}{L-1} \tag{B.8}$$

Se o cruzamento é executado por uma escolha aleatória, com uma probabilidade $\boldsymbol{P_c}$, a probabilidade de sobrevivência pode ser reescrita como:

$$P_s \geq 1 - P_c . \frac{\delta(H)}{L-1} \tag{B.9}$$

O efeito combinado do operador de seleção e cruzamento é obtido pela multiplicação do número de esquemas esperados para a seleção pela probabilidade de sobrevivência após o cruzamento, representado na seguinte equação:

$$m(H, t+1) \geq m(H, t) . \frac{f(H)}{\bar{f}} . (1 - P_c . \frac{\delta(H)}{L-1}) \tag{B.10}$$

O último operador a ser considerado é a mutação. Considerando a mutação como a alteração aleatória de uma única posição com probabilidade $\boldsymbol{P_m}$. Para a sobrevivência do esquema $\boldsymbol{H}$, todas as posições fixadas têm que sobreviver. Sendo a sobrevivência de uma posição dada por $(1 - P_m)$ e que cada uma das mutações é estaticamente independente, um determinado esquema sobrevive quando cada uma das o(H) posições fixadas do esquema sobrevivem. Multiplicando a probabilidade de sobrevivência $(1 - P_{\boldsymbol{m}})$ por ela mesmo o(H) vezes, temos a probabilidade de mutação dada por $(1 - P_m)^{o(H)}$. Para valores pequenos de $\boldsymbol{P_m}$ ($P_m << 1$) a probabilidade de sobrevivência do esquema pode ser aproximada por:

$$P_s \cong (1 - o(H) . P_m) \tag{B.11}$$

Assim, temos que o número de esquemas esperado, na próxima geração, após a atuação dos operadores seleção, cruzamento e mutação é dada por:

$$m(H,t+1) \geq m(H,t).(1 - P_c.\frac{\delta(H)}{L-1} - o(H).P_m).\frac{f(H)}{\bar{f}} \qquad \text{(B.12)}$$

Da Equação (B.12) concluímos que esquemas curtos, de baixa ordem e acima da média da população, apresentam um aumento no seu número de uma geração para outra. Esta equação representa o Teorema Fundamental dos Algoritmos Genéticos e demonstra que com a evolução do processo de busca existe uma convergência dos esquemas presentes na população.

Referências Bibliográficas

BOECHEL, T., 2003, *Algoritmo de Otimização: Uma Abordagem Híbrida Utilizando o Algoritmo das Formigas e Genético*, Tese de M.Sc., UFSC, Florianópolis, SC, Brasil.

BONABEAU, DORIGO, M., THERAULAZ G., 1999, "Swarm Intelligence: From Natural to Artificial Systems", Oxford University Press, New York.

BURIOL, L.S., 2000, *Algoritmo Memético para o Problema do Caixeiro Viajante Assimétrico como parte de Framework para Algoritmos Evolutivos*, Tese de M.Sc., DENSIS/FEE/UNICAMP, Campinas, SP, Brasil.

CANTÚ-PAZ, E., 1998, "A Survey of Parallel Genetic Algorithms", *Calculateurs Paralleles*, v. 10, n.2.

CANTÚ-PAZ, E., 1999, *Topologies Migration Rates, and Multi-population Parallel Genetic Algorithms*. In: IlliGAL Report Nº 99007, Illinois Genetic Algorithms Laboratory, University Illinois at Urbana-Champaign, Urbana.

CARDIERI, M. A. C. A., 1998, Agentes Inteligentes: Noções Gerais, Monografia, DENSIS/FEE/UNICAMP, Campinas, SP, julho.

CHAPOT, J.L.C., 2000, *Otimização Automática de Recargas de Reatores à Água Pressurizada Utilizando Algoritmos Genéticos*, Tese de D.Sc., Programa de Engenharia Nuclear, COPPE/UFRJ, Rio de Janeiro, Brasil.

CLARO, L.H., 1992, *O método dos Psudo-Harmônicos: Uma Nova Opção Usando Discretização Nodal*. Tese de D.Sc., COPPE/UFRJ, Rio de Janeiro, Brasil.

COLORMI, A., DORIGO, M., MANIEZZO, V., 1991, "Distributed Optimization by Ant Colonies", Proceedings of ECAL91 – European Conference on Artificial Life, Paris, France, F. Varela and P. Bourgine (Eds.), Elsevier Publishing, pp. 134-142.

DARWIN, C.R., 1859, On the Origin of Species by Means of natural Selection. London, John Murray.

DECHAINE, M.D., FELTUS, M.A., 1995, *Nuclear Fuel Management Optimization Using Genetic Algorithms*, Nuclear Technology, v. III, pp. 109-114.

DE LIMA, A, M.M., 2000, *Modelo de Ilhas para a Implementação Paralela do Algoritmo Evolucionário de Otimização PBIL*, Tese de M.Sc., Programa de Engenharia Nuclear, COPPE/UFRJ, Rio de Janeiro, Brasil.

DENEUBOURG J.L., PASTEELS J.M., VERHAEGHE J.C., 1983, "Probabilistic behaviour in ants: a strategy of errors", J. Theor. Biol. 105, 259-271.

DENEUBOURG J.L., ARON S., GOSS S., PASTEELS, J.M., DUERINCK G., 1986, "Random behavior, amplification processes and number of participants: how they contribute to the foraging properties of ants", Physic 22D, 176-186.

DORIGO, M., GAMBARDELLA, L.M., 1997, "Ant Colony System: A cooperative Learning Approach to Traveling Salesman Problem", *IEEE Transactions and Evolutionary Computation*, v.1, n.1, pp. 53-66.

DORIGO M., DI CARO G, SAMPELS M., 2002, Ants 2002 – "From Ant Colonies to Artificial Ants", Third International Workshop on Ant Algorithms. Brussels, Belgium.

FRANKLIN, S, GRAESSER, A., 1996, "It is an Agent or Just Program? A Taxonomy for Autonomous Agents". *Proceedings of third International Workshop on Agents Theories, Architetures and Languages*, Springer-Verlag.

GALPERIN, A., 1995, "Exploration of the Search Space of the In-Core Fuel Management Problem by Knowledge-Based Techniques", *Nuclear Science and Engineering*, v. 119, pp. 144-152.

GAMBARDELLA, L.M., DORIGO, M., 1996, "Solving Symmetric and Asymmetric TSPs by Ant Colonies", *IEEE Conference on Evolutionary Computation*, ICEC'96, Nagoya, Japan.

GAMBARDELLA, L.M., TAILLARD, E.D., DORIGO, M., 1999, "Ant Colonies for the Quadratic Assignment Problem", *Journal of the Operational Research Society*, v.50, pp.167-176.

GOLDBARG, M.C., LUNA, H., Pacca L., 2000, "Otimização combinatória e programação linear: modelos e algoritmos", Rio de Janeiro: Editora Campus.

HAYKIN, S., 1994, Neural Networks – A comprehensive foundation, Macmillan College Publishing Company Inc., New York.

HOLLAND, J.H., 1975, *Adaptation In Natural and Artificial Systems*, Ann Arbor, University of Michigan Press.

KAELBLING, L.P., LITTMAN, L.M., MOORE, A.W., 1996, "Reinforcement Learning: a Survey", Journal of Artificial Intelligence Research, v.4, pp. 237-285.

KAUFMAN, S., 1995, "At Home In The Universe: The Search For Laws Of Self-Organization And Complexity", Oxford University Press.

KIRKPATRICK, S., GELLAT, C.D, VECCHI, M.P., 1983, *Optimization by Simulated Annealing, Science*, v. 220, pp. 671-680.

KROPACKZEK, D.J., TURINSKY, P.J., 1991, *In-core Nuclear Fuel Management Optimization for Pressurized Water Reactors Utilizing Simulated Annealing*, Nuclear Technology v.95, n.9, pp.9-31.

LANGENBUCH, S., MAURER, W., WERNER, W., 1977, *Coarse-Mesh Flux Expansion Method for Analysis of Space-Time Effects in Large Water Reactor Cores*, Nucl. Sci. Eng., 63, pp.437-456.

MACHADO, L., 2001, *Otimização da Recarga do Combustível Nuclear por Agentes Artificiais*, Tese de D.Sc., Programa de Engenharia Nuclear, COPPE/UFRJ, Rio de Janeiro, Brasil.

MAES, P., 1995, "Artificial Life Meets Entertainment: Life Like Autonomous Agents", *Communications of the ACM*, 38,11, 108-114.

MIDDENDORF, M., REISCHLE, F., SCHMECK, H., 2000, "Information Exchange in Multi Colony Ant Algorithms", From Institut Für Angewandte Informatik", Universität Karlsruhe, Karlsruhe, Germany.

MINSKY, M., 1994, "A Conversation with Marvin Minsk About Agents", *Communications of the ACM*, v.37, n.7.

MONTAGNINI, B., SORAPERRA, P., TRENTAVIZI, C., SUMINI, M., ZARDINI, D. M., 1994, *A Well-Balanced Coarse-Mesh Flux Expansion Method*, Ann. Nucl. Energy, v.21, n.11, pp. 45-53.

POON, P.W., PARKS, G.T., 1992, *Optimizing PWR Reload Core Design*, Parallel Problem Solving from Nature, v. 2, pp. 371-380.

REINELT, G., *TSPLIB – A Library of Traveling Problem and Related Problem Instances*, 2005, http://www.iwr.uni-heidelberg.de/groups/comopt/software/TSPLIB95/, biblioteca com rotas de problemas do caixeiro viajante.

RUSSEL, S.J., NORVIG, P., 1995, *Artificial Intelligence: A Modern Approach*, Englewood Cliffs, NJ: Pretince Hall.

SILVA, F.M.M.C., 1998, "Mini curso sobre Arquitetura de Agentes Inteligentes". http://www.di.ufpe.br/~fmmcs/agentes/resumo.html.

STUTZLE, T., DORIGO, M., 2000, *ACO Algorithms for the Traveling Salesman Problem*, IRIDIA, Université Libre de Bruxelles, Belgium.

WOOLDRIDGE, M., JENNINGS, P.S, 1995, "Agents Theories, Architectures, and Languages: a Survey, "in Wooldridge and Jennings Eds., *Intelligent Agents*, Berlin: Springer-Verlag, 1-22.